

Faculté de génie
Génie électrique et génie informatique

Méthodologie d'évaluation d'utilisabilité de logiciel

Mémoire de maîtrise es sciences appliquées
Spécialité : génie électrique

Miriam PEREZ BERRO

Sherbrooke (Québec) Canada Mars 2007

IV - 1815



Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-37923-3
Our file Notre référence
ISBN: 978-0-494-37923-3

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.



TABLE DES MATIÈRES

Table des figures

Liste des tableaux

RÉSUMÉ

L'un des problèmes des équipes de développement de logiciel consiste à évaluer la facilité d'utilisation de leurs produits, c'est-à-dire, à évaluer la qualité de l'interaction personne-machine. Le développeur lui-même n'est pas capable, dans la plupart des cas, de juger la qualité d'utilisation de l'interface qu'il propose, pour la simple raison qu'il n'est pas l'utilisateur réel du produit. Une fois les erreurs de code éliminées, il est possible que les utilisateurs d'un logiciel ne puissent pas encore travailler efficacement à cause d'erreurs de conception d'utilisabilité. Il s'avère nécessaire de disposer d'une méthodologie d'évaluation dès le départ d'un projet logiciel.

Il n'existe pas d'outils d'évaluation génériques applicables à tout type de logiciel. Il est nécessaire de développer une méthodologie appropriée au profil de la communauté d'utilisateurs et des tâches qu'ils veulent automatiser à partir de leur utilisation. Cette méthodologie doit aussi tenir compte des caractéristiques de l'environnement de développement des produits.

Le logiciel ciblé dans ce projet est un système d'aide à la construction d'horaires[1] utilisée dans un environnement de travail administratif. Le système comprend une application autonome (le logiciel appelé Diamant) et une application web (le logiciel appelé Diamant Web). Une évaluation sommative a été réalisée étant donné que le système sous évaluation avait été déjà livré. Celle-ci a été appliquée au sein du groupe de recherche en génie logiciel εχit[2]. Il s'agit d'une petite équipe de développement de logiciel avec des ressources limitées où le but est de créer un produit qui marche selon les exigences le plus rapidement possible. Le système ciblé exécutait correctement les fonctionnalités demandées, mais les utilisateurs réels le trouvaient difficile à utiliser. Les résultats d'une évaluation d'utilisabilité devaient être obtenus rapidement et au plus bas coût possible. Cela conduit à proposer une méthodologie simple et légère.

L'approche utilisée afin de mener au terme une évaluation d'utilisabilité est basée

[1]Scheduling - Timetables
[2]Département de génie électrique et de génie informatique de l'Université de Sherbrooke

sur la prise et l'analyse de données de terrain et la construction de prototypes d'évaluation d'utilisabilité.

La contribution principale du présent travail comprend un ensemble de documents, modèles de guides, de maquettes d'interaction, de rapports et d'analyse d'interfaces conçus et construits à l'occasion de l'évaluation d'utilisabilité du système ciblé. Ces documents et ces modèles sont des exemples applicables à n'importe quel logiciel dont les caractéristiques et l'analyse contextuelle seraient similaires aux logiciels évalués. En outre, la méthodologie constitue elle-même une contribution en raison des contraintes de temps et de coût imposées.

Les résultats obtenus comprennent des lignes directrices simples pour mettre en œuvre l'évaluation d'utilisabilité dans un projet logiciel, les exemples de suggestions et de guides afin d'améliorer l'interaction personne-machine présentée par le logiciel sous évaluation.

Phrases et mots clés :

Usabilité, utilisabilité, convivialité, facilité d'utilisation, qualité d'utilisation de logiciel, interaction personne-machine, interaction homme-machine, interface personne ordinateur, conception de l'interface d'utilisateur

CHAPITRE 1
INTRODUCTION

Le domaine de recherche sur l'interaction homme-machine (HCI[1]) a adopté et adapté les idées à propos des modèles mentaux de l'utilisateur afin d'aller plus loin dans l'étude de sa principale aire d'application : l'utilisabilité.

Le modèle mental des humains a été étudié par les sciences cognitives comme une partie des efforts pour comprendre comment les humains connaissent, perçoivent, prennent des décisions et construisent des comportements dans une grande diversité d'environnements. Un modèle mental est une explication du processus de la pensée de quelqu'un en ce qui concerne comment quelque chose fonctionne dans le monde réel. Il s'agit d'un type de symbolisme ou représentation interne de la réalité externe, en rapport avec la connaissance. Cette approche provenant de la psychologie est discutée aujourd'hui dans le domaine de l'interaction homme-machine et l'utilisabilité.

Les sections suivantes sont consacrées à présenter les concepts liés à l'utilisabilité, le but du projet et l'organisation générale du document.

1.1 Concept d'utilisabilité

La définition d'utilisabilité se résume comme la capacité d'un système à permettre à ses utilisateurs de faire effectivement et efficacement ce pour quoi ils l'utilisent.

L'utilisabilité peut faire la différence entre le succès ou l'échec d'un produit. Ceci sert à stimuler les recherches dans ce domaine.

Les pratiques de programmation telles que la programmation extrême XP[2][2], les méthodes agiles [6][18] et la programmation pragmatique[14] mettent en relief l'importance de l'utilisateur dès le départ d'un projet logiciel. Toutes les analyses et les évaluations d'uti-

[1]Human-Computer Interaction
[2]Extreme Programming

lisabilité sont favorisées par le design itératif fait en collaboration avec l'utilisateur et contribuent directement aux améliorations envisagées dans chaque itération.

Les guides pour être un programmeur pragmatique remarquent que l'idéal serait de devenir un utilisateur pour mieux comprendre ses propres besoins. Dans la pratique, travailler avec l'utilisateur pour essayer de penser comme lui, permet de connaître mieux les spécifications et d'établir les bases de la communication active avec ce dernier. Également, un rôle prépondérant est attribué à l'évaluation d'utilisabilité avec l'utilisateur réel, sous condition de fonctionnement réel et orienté aux facteurs humains, ainsi que la vérification des interfaces graphiques d'utilisateur avec des outils spécialisés. Le concept d'utilisabilité va être approfondi dans les paragraphes suivants.

Le terme utilisabilité est la francisation du mot en anglais "usability", employé pour définir le caractère d'un logiciel à être utilisable ou fonctionnel. Convivialité, facilité d'utilisation, facilité d'emploi, ergonomie de logiciel sont d'autres termes ou phrases utilisés pour traduire le terme anglais "usability". D'une part, l'utilisabité représente la qualité d'un matériel ou d'un logiciel qui est facile et agréable à utiliser et à comprendre, même par quelqu'un qui a peu de connaissance en informatique et/ou en ordinateurs. D'autre part, dans le cas des logiciels utilisés pour travailler, l'utilisabilité signifie la transparence du logiciel en tant qu'outil[3], qui permet à la personne d'avancer dans son travail et d'augmenter sa productivité, sans introduire des contraintes ou des obstacles additionnels à sa tâche. Finalement, bien que souvent utilisé avec le sens de "convivialité", le terme utilisabilité possède un sens plus large qui correspond à la capacité d'un système à permettre à ses utilisateurs de faire efficacement ce pour quoi ils l'utilisent. Afin que le travail soit fait, le système utilisable doit non seulement être facile à utiliser, mais aussi fiable et efficace.

L'utilisabilité est l'un des domaines de l'interaction personne-ordinateur. Elle diminue l'écart entre les gens et les machines. L'interface d'utilisateur ou interface humain-ordinateur se réfère à la partie du matériel et/ou du logiciel qui permet à une personne de communiquer avec un ordinateur. Cela inclut les dispositifs d'entrée (la façon de parler de la personne avec la machine) et de sortie (la façon de parler de la machine avec la personne). Chacun de ces composants d'interface possèdent des dispositifs correspondants aux chaînes visuelles, audio et tactiles du cerveau humain. L'utilisabilité étudie ces éléments de l'expérience de

[3]Tout au long de ce document, les logiciels sont supposés des produits utilisés dans un environnement de travail.

l'utilisateur.

L'analyse de l'utilisabilité d'un logiciel doit permettre de répondre aux questions suivantes :

1. Qui sont les utilisateurs ? Quelles sont les caractéristiques de ceux qui arrivent à utiliser le produit étudié ?
2. Que recherchent les utilisateurs ? Quel est le but des utilisateurs ?
3. Le logiciel est-il facile ou difficile à utiliser ?

Cela peut paraître simple et évident, mais la réalité montre souvent que la majorité des concepteurs de logiciel ne présentent pas l'information du point de vue de l'utilisateur, et ils n'ont pas d'idée précise sur les buts poursuivis par les utilisateurs réels.

Une évaluation d'utilisabilité est un examen visant à établir si le système mis en œuvre remplit ses fonctions premières telles que déterminées par ses utilisateurs. Il s'agit d'une procédure qui fournit une portrait de ce que les utilisateurs recherchent et qui porte sur ce que les utilisateurs arrivent réellement à faire avec le logiciel. Il comprend :

1. Le choix des utilisateurs parmi la communauté des utilisateurs.
2. L'observation 1 à 1 pendant l'utilisation du logiciel.

Une grande partie de l'évaluation consiste à observer et à encourager le participant à exprimer ce qu'il pense pendant la réalisation de tâches prédéterminées afin d'identifier les freins réels rencontrés.

Initialement, les laboratoires d'utilisabilité[4] étaient des laboratoires d'ergonomie. L'ergonomie est définie comme étant "l'ensemble des études et des recherches sur l'organisation du travail et l'aménagement de l'équipement en fonction des possibilités de l'homme". L'ergonomie s'intéresse à la meilleure adéquation physique possible entre les équipements et leurs utilisateurs et la meilleure adéquation "intellectuelle" entre la conception du poste de travail et l'individu occupant ce poste. Actuellement, elle comprend aussi l'équipement et les logiciels nécessaires afin de réaliser des tests d'utilisabilité de logiciel pour évaluer sa qualité du point de vue de l'interface d'utilisateur.

[4]Usability laboratory

1.2 But du projet

L'objectif du projet a été de déterminer les concepts de base, d'explorer les principales techniques de mesure et de générer un premier ensemble d'outils afin d'évaluer l'utilisabilité des produits logiciels. L'application des outils crées a permis d'identifier les problèmes d'utilisabilité d'un système logiciel. La méthodologie suivie pour atteindre cet objectif a respecté certaines contraintes de coût et de temps. Les étapes du parcours sont indiquées dans le tableau 1.1. Plus de détails sont présentés dans le chapitre 3, section 3.5.

Tableau 1.1 – Étapes du projet

Item	**Activité**
1	Recherche bibliographique
2	Définir un sous-ensemble de métriques et de techniques d'intérêt pour le projet
3	Valider ce sous-ensemble de techniques et construire des outils
4	Évaluer un cas d'utilisation

Les sources fondamentales du présent travail résident dans les standards de qualité de logiciel ISO9126 et particulièrement, dans les normes ISO9241 de guide sur utilisabilité. La figure 1.1 montre la définition ISO d'utilisabilité d'un système.

Les références principales se trouvent dans les livres [4],[17],[1],[5],[8] et [24] et dans les articles [22],[13],[16],[25],[9],[12],[15],[11] et [20]

1.3 Organisation du document

Le document est organisé selon le détail ci-dessous.

D'abord, ce chapitre a présenté la définition du concept d'utilisabilité qui sert de point de départ de ce travail.

Le chapitre 2 présent une analyse de comment l'utilisabilité est perçue depuis plusieurs perspectives. Son importance du point de vue des utilisateurs, des gestionnaires et des développeurs est traitée. La place que l'utilisateur prend dans les méthodes actuelles de développement de logiciel est aussi considérée.

Ensuite, les composants principaux d'évaluation d'utilisabilité sont explorés dans le

Figure 1.1 – L'utilisabilité selon le standard ISO9241

ISO 9241-11

Un système est utilisable lorsqu'il permet à

l'utilisateur

de réaliser sa tâche avec

efficacité, efficience et satisfaction

dans le contexte d'utilisation spécifié

chapitre 3. Ils comprennent essentiellement les métriques, les audits et les normes standards de qualité d'utilisabilité.

Le chapitre 4 est consacré à l'étude théorique des méthodologies d'évaluation d'utilisabilité.

Le chapitre 5 expose la méthodologie développée dans le présent projet.

Les éléments guides de la conception de l'interface d'utilisateur pris pour l'évaluation ainsi que l'analyse du profil utilisateur-tâches est abordé dans le chapitre 6

Le chapitre 7 expose les résultats obtenus de l'évaluation d'utilisabilité faite sur le système logiciel Diamant développé par le groupe de recherche εχit cité auparavant.

Les annexes présentent les livrables réalisés dans le projet. L'annexe A montre les modèles de documents destinés aux différents participants d'une évaluation d'utilisabilité, l'annexe B les modèles de rapports et l'annexe C les outils générés pendant la réalisation de ce travail.

Cette structure de présentation a été choisie de façon à justifier les méthodes appliquées et à exposer la méthodologie développée d'abord et, ensuite, à montrer l'ensemble

de livrables du projet.

CHAPITRE 2
IMPORTANCE DE L'UTILISABILITÉ

Ce chapitre expose l'importance de l'utilisabilité d'un logiciel, d'abord du point de vue des concepteurs et développeurs, des gestionnaires et des utilisateurs. Ensuite, la place que les techniques actuelles de développement de logiciel assignent aux utilisateurs est analysée.

2.1 L'utilisabilité du point de vue des concepteurs et des développeurs

Les concepteurs et les développeurs, face aux problèmes d'utilisabilité, devraient se poser la question suivante : sommes-nous vraiment centrés sur le client[1] ? sommes-nous en train de faciliter l'utilisation de notre produit aux utilisateurs ?

En effet, la première étape pour faire face à une analyse d'utilisabilité consiste à être sûr que le processus de développement est centré sur l'utilisateur. Cela signifie que les utilisateurs sont impliqués dans le processus de conception et d'évaluation du système.

L'évaluation proposée aide à mesurer la situation actuelle du logiciel et fournit une structure visant son amélioration. Elle permet d'identifier les points forts et les points faibles dans le processus actuel de développement.

L'utilisabilité du point de vue du développeur peut signifier la réussite ou l'échec d'un produit. C'est pourquoi cet aspect du produit ne doit pas être négligé.

Un développeur doit être conscient des points suivants et agir en conséquence :

1. La façon de faire des utilisateurs est presque impossible à prédire.
2. Les problèmes qui les utilisateurs rencontreront sont très difficiles à prédire.
3. Les problèmes de l'utilisateur peuvent être inconcevables du point de vue des concepteurs et des développeurs.

[1]Customer centered

4. La confiance de l'utilisateur est l'un des éléments les plus fragiles dans l'interaction homme-machine.

2.2 L'utilisabilité du point de vue des gestionnaires

Il est fréquent d'essayer de convaincre les gestionnaires (patrons des développeurs) de la nécessité d'investir dans l'utilisabilité en faisant valoir des arguments sortis de l'analyse coût-avantages. La qualité de ce facteur peut augmenter les revenus, créer des clients fidèles, améliorer la valeur de la marque et les résultats du processus interne de gestion et de développement. Une conception qui néglige l'utilisabilité, du point de vue de la gestion, réduit la productivité des efforts de travail, augmente les délais et conditionne en grande mesure la réussite ou l'échec d'un système.

Cependant, il est aussi fréquent de constater que les arguments sont écoutés et après, ils sont pratiquement ignorés.

La raison principale est que la décision sur l'incorporation de l'utilisabilité dans une organisation ne dépend pas d'un argument logique. La conception et la planification centrée sur l'utilisateur ont un impact sur chaque partie de cette organisation. Les décisions par rapport à l'utilisabilité entraînent des changements de type organisationnel. Ces changements passent à travers du changement de mentalité des gens.

2.3 L'utilisabilité du point de vue des utilisateurs

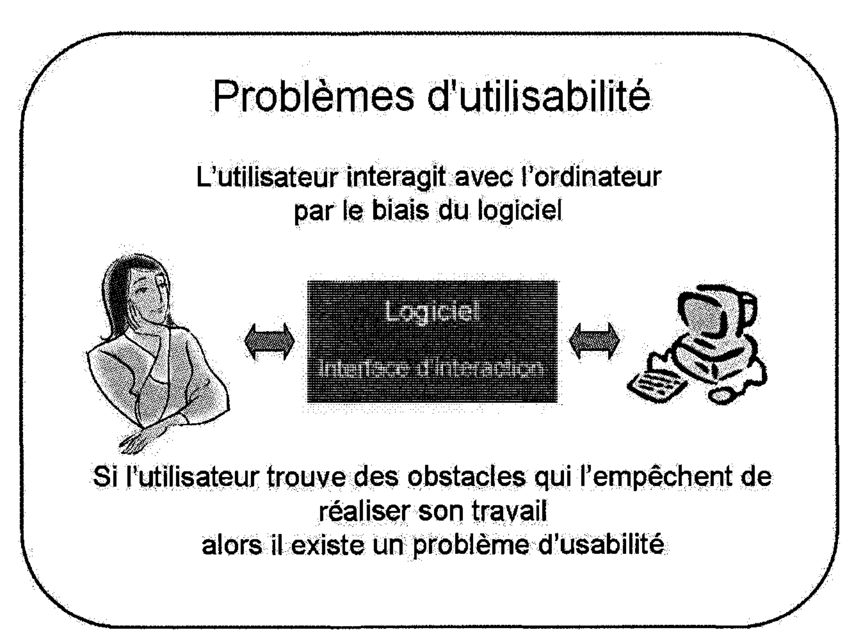
L'utilisabilité s'intéresse à la relation entre les outils et leurs utilisateurs. Pour qu'un outil soit effectif, il doit permettre à l'utilisateur prévu, d'accomplir les tâches de la meilleure façon possible. Le même principe s'applique aux ordinateurs, sites web et d'autres logiciels. Ces systèmes fonctionnent vraiment lorsque leurs utilisateurs sont capables de les employer efficacement.

Du point de vue de l'utilisateur, ce qui fait qu'un logiciel soit plus ou moins utilisable dépend de plusieurs facteurs :

1. Dans quelle mesure le logiciel remplit les besoins des utilisateurs.
2. Dans quelle mesure le navigation à travers l'application s'ajuste aux tâches de l'utilisateur.

3. Dans quelle mesure l'application s'ajuste aux attentes de l'utilisateur.

Figure 2.1 – Interaction personne-machine



Le développeur plus perspicace, peut seulement créer un système hautement utilisable par le biais d'un processus qu'implique activement l'obtention d'information de la part des gens qui réellement utilisent le système.

Bref, l'utilisabilité est la qualité d'un système qui le fait :

1. Facile à apprendre.
2. Facile à utiliser.
3. Facile à se rappeler de la façon de l'utiliser.
4. Tolérante à l'erreur.
5. Subjectivement agréable, ni frustrant, ni irritant.

Ces cinq points constituent les paramètres mesurables de la qualité d'une interface d'utilisateur.

L'importance de l'utilisabilité depuis la perspective de l'utilisateur réside dans le fait qu'elle détermine la différence entre réaliser une tâche complètement et exactement ou non, et d'avoir une expérience utilisateur positive ou de se sentir frustré, même irrité.

Figure 2.2 – But de l'utilisateur

Le but de l'utilisateur

Réaliser son travail efficacement

1. rapidement, à temps
2. sans erreurs, soigneusement
3. sans s'irriter
4. sans se sentir bloqué

L'ordinateur doit être un outil

L'ordinateur ne doit pas ajouter des difficultés

2.4 Arriver à un haut niveau d'utilisabilité

La clé principale pour maximiser l'utilisabilité consiste à mettre en pratique la technique de conception et de développement itératif, où le raffinement progressif par le biais de l'évaluation a lieu dès le départ. Les étapes d'évaluation habilitent les concepteurs et les développeurs à incorporer la rétroaction de l'utilisateur et du client jusqu'à ce que le système atteigne un niveau acceptable d'utilisabilité.

La méthode préférée pour assurer l'utilisabilité consiste à faire tester une version du système fonctionnel par plusieurs utilisateurs réels. Atteindre un haut niveau d'utilisabilité demande la focalisation des efforts de conception sur l'utilisateur final du système. Il existe plusieurs façons de déterminer quel est le profil de l'utilisateur primaire, comment il travaille et quel type de tâches il doit accomplir. Toutefois, l'agenda et le budget consacré au développement peuvent quelquefois empêcher l'approche idéale. Des méthodes alternatives incluent le test sur des prototypes du système, des inspections d'utilisabilité conduites par des experts et des modèles cognitifs.

2.5 Le point de départ pour l'évaluation d'utilisabilité

La plupart des développeurs n'ont jamais vu à l'utilisateur utiliser leur produit logiciel. La première expérience peut entraîner un choc. Il est possible de trouver des utilisateurs qui n'arrivent pas à utiliser le logiciel, spécialement certaines fonctionnalités. Le premier type d'évaluation à réaliser consiste à observer l'utilisateur lorsqu'il travaille avec le logiciel sous évaluation.

Comment entreprendre une évaluation avec l'utilisateur ? Il existe un petit nombre de règles et d'étapes à suivre. Elles sont :

1. Choisir une personne représentative de l'utilisateur final du logiciel. Il faudrait essayer de trouver quelqu'un qui n'ait jamais travaillé avec lui, mais très proche de l'utilisateur futur du logiciel.
2. Établir le scénario, placer la personne face à l'ordinateur, lui donner les instructions nécessaires et lui demander de penser à haute voix, spécialement lorsqu'elle est en train d'élucider quelque chose.
3. Après ne pas parler, être en silence, ne pas perturber l'espace de l'utilisateur.
4. Observer ce que fait la personne. Si la personne pose une question, il faut lui expliquer qu'elle doit agir comme si elle était seule.
5. Prendre note de tous les détails observés et de ce qui posse des problèmes à l'utilisateur.
6. Ensuite, lui demander ce qu'elle a pensé en termes généraux du logiciel comme outil pour réaliser la tâche donnée.

2.6 Utilisabilité, utilisateurs et programmation extrême

L'eXtreme Programming (XP)[2][3] est l'un des principaux représentants des méthodes de développement appelées "agiles" qui mettent l'accent sur le travail d'équipe et la réactivité. XP définit un processus de développement logiciel, c'est-à-dire un ensemble de techniques destinées à organiser le travail d'une équipe de développement.

Ces pratiques se concentrent sur la construction proprement dite du logiciel, après des études d'opportunité et de faisabilité d'un projet.

D'une part, dans toute équipe de développement, il faut tenir compte de l'économie et de l'efficacité.

D'autre part, la production d'un logiciel de qualité et la satisfaction des besoins des clients ne sont pas fréquemment en accord avec les principes de l'économie et de l'efficacité du point de vue commercial.

XP propose une réponse à ce dilemme, qui s'appuie sur les principes suivants :

1. L'utilisateur dirige lui-même le projet de très près grâce à des cycles itératifs extrêmement courts (1 ou 2 semaines). Ainsi, l'utilisateur fait partie de l'équipe de développement.
2. L'équipe livre très tôt dans le projet une première version du logiciel, et les livraisons de nouvelles versions s'enchaînent ensuite à un rythme soutenu pour obtenir une rétroaction maximale des utilisateurs sur l'avancement des développements.
3. L'équipe s'organise elle-même pour atteindre ses objectifs, en favorisant une collaboration maximale entre ses membres.
4. L'équipe met en place des tests automatiques pour toutes les fonctionnalités qu'elle développe, ce qui garantit au produit un niveau de robustesse très élevé.
5. Les développeurs améliorent sans cesse la structure interne du logiciel pour que les changements produits par les évolutions y restent faciles et rapides.

2.7 Utilisabilité, utilisateurs et le programmeur pragmatique

Dans la section suivante, nous allons passer revue les aspects plus importants du livre[14] à propos de l'importance de l'utilisateur dans le processus de développement de logiciel.

Le programmeur pragmatique s'intéresse au cœur du processus de développement de logiciel : à partir des exigences, il produit un code performant, robuste et qui fait plaisir à ses utilisateurs.

Étant donné que le produit logiciel n'est pas parfait, les auteurs introduisent le concept flou de "logiciel assez bon". La question immédiate est : "assez bon" pour qui ? La réponse est "assez bon" pour les utilisateurs en premier lieu, pour l'équipe de maintenance et pour les développeurs en dernier lieu.

L'effet obtenu est l'augmentation de la productivité et de la satisfaction subjective des utilisateurs (acheteurs, clients, etc.). Il vaut mieux d'avoir un bon logiciel aujourd'hui qu'un logiciel parfait demain. La clé pour atteindre ce but consiste à impliquer l'utilisateur dans le projet. Il faut se demander quel est le degré de qualité que doit remplir un logiciel selon les exigences de l'utilisateur. Ignorer les spécifications, les exigences, les contraintes de l'utilisateur, ce n'est pas professionnel.

La communication occupe une place déterminante dans le processus de développement de logiciel. En principe, une bonne idée ne peut pas se développer sans une communication efficace des gens. Il faut communiquer avec les utilisateurs pour comprendre leurs besoins. Il faut communiquer avec d'autres développeurs par le biais de la documentation du code. Il faut communiquer avec l'équipe de travail au complet.

Un haut niveau d'abstraction, une concentration sur la résolution du problème, en ignorant les détails d'implémentation favorise le processus de communication. Les langages ont de l'influence sur la façon de penser et de communiquer. Le langage du domaine du problème peut suggérer une solution de programmation. Alors, une bonne pratique de programmation est d'utiliser le langage du domaine de l'application pour écrire le code. Une autre technique consiste à inventer un mini langage qui exprime les idées des utilisateurs tels qui elles sont, un langage qui sert à capturer les exigences des utilisateurs. L'utilisation de ce langage, peut conduire à une codification très proche du domaine de l'application et par conséquent, du problème à résoudre.

Le domaine des utilisateurs se divise en utilisateurs finaux et utilisateurs secondaires. Les premiers connaissent les règles d'affaires et les sorties requises. Les deuxièmes comprennent l'équipe d'opérations, les administrateurs de configuration et de test, les programmeurs de support et de maintenance, et finalement, les développeurs.

En ce qui concerne l'estimation de la planification d'un projet, les éléments à considérer sont la liste des exigences, l'analyse des risques, les étapes de conception, d'implémentation, d'intégration et de validation avec les utilisateurs. Il est nécessaire d'itérer et de raffiner la planification avec le code et tenir compte qu'elle dépend de la productivité de l'équipe de travail et de l'environnement.

L'analyse des exigences marque le commencement du travail dans un projet. Les exigences traduisent les besoins des utilisateurs. Une exigence est quelque chose qui doit être

accomplie, plus précisément, une obligation qu'un système ou un processus doit respecter pour satisfaire un besoin suffisamment important pour justifier des coûts de développement et d'exploitation. L'analyse est complexe parce que les exigences ne sont pas habituellement claires. Réunir des exigences n'est pas suffisant, il est nécessaire d'aller plus profondément pour les trouver. Également, il est important de découvrir la raison sous-jacente de pourquoi l'utilisateur fait ce qu'il fait, au lieu de seulement voir ce qu'il fait (les étapes, la manière, etc.). Un autre aspect est de distinguer parmi les exigences, la politique (règles d'affaires) et l'implémentation. L'idéal serait de devenir un utilisateur pour comprendre mieux ses besoins. Dans la pratique, travailler avec l'utilisateur pour essayer de penser comme lui, permet de connaître mieux les spécifications et d'établir les bases de la communication active avec l'utilisateur.

Dans le but d'assurer la consistance de la nomenclature parmi l'audience, il est convenable de maintenir un glossaire du projet en utilisant les termes propres au domaine des utilisateurs et de l'application. Il faut éliminer les ambiguïtés dans la terminologie de l'application et de l'utilisateur, causées par la signification courante ou familière de certains mots. Le glossaire doit être amplement accessible pour les utilisateurs, les développeurs, le groupe de support, etc.

En ce qui concerne les tests, deux points fondamentaux sont dans la liste :

1. Le test d'utilisabilité avec l'utilisateur réel et dans des conditions de fonctionnement réel, dirigé aux facteurs humains
2. La vérification des interfaces graphiques d'utilisateur avec des outils spécialisés.

Par rapport aux attentes des utilisateurs, une application est réussite si elle implémente ses spécifications correctement. En réalité, la réussite d'un projet est mesurée selon les attentes des utilisateurs. Excéder légèrement les attentes des utilisateurs est avantageux. Les auteurs encouragent le fait de communiquer des attentes et de travailler avec l'utilisateur pendant le processus de développement pour s'assurer qu'il comprend exactement le produit à livrer. Également, ils rappellent de ne pas perdre de vue les problèmes d'affaire que le produit tente de résoudre. Il faut savoir comment gérer les attentes, c'est-à-dire, suivre activement ce que les utilisateurs attendent du système et les intégrer dans le processus pour atteindre une complète compréhension. L'équipe construit quelque chose que l'utilisateur peut voir et essayer en utilisant les techniques de prototypage.

CHAPITRE 3
ÉVALUATION D'UTILISABILITÉ

Une évaluation d'utilisabilité est une analyse qualitative et quantitative de l'utilisabilité d'un prototype ou d'un système. Le but est de fournir une retroaction (feedback en anglais) de l'utilisateur dans le cadre de promouvoir un processus itératif de développement. L'évaluation d'utilisabilité aide les concepteurs à identifier et à comprendre les problèmes, à trouver les causes sous-jacentes et à planifier les changements nécessaires dans le logiciel pour le corriger. Les aspects principaux de l'évaluation d'utilisabilité tels que les métriques, les audits, l'automatisation et les standards sont parcourus dans les sections suivantes.

3.1 Métriques d'utilisabilité

Les métriques ou marques de référence d'utilisabilité sont des mesures qui aident en tant que guides du niveau d'utilisabilité d'un produit. Ces variables mesurables comprennent :

1. Le temps pris par l'utilisateur pour réaliser une tâche
2. Le taux des erreurs fait par l'utilisateur ou la productivité (une mesure du nombre des actions correctement réalisées par l'utilisateur)
3. La période d'apprentissage de comment faire un ensemble de tâches de base
4. La capacité de rétention dans la mémoire de comment faire les tâches de base
5. La satisfaction subjective exprimée par l'utilisateur

Les métriques [26] [23] [7] [19] incluent les caractéristiques montrées dans le tableau 3.1.

3.2 Audit d'utilisabilité

L'audit d'utilisabilité[26] comprend les éléments suivants :

1. Fonctionnalité : Existe-t-il une raison qui force à l'utilisation du logiciel ? Le logiciel est-il fiable ? Il s'agit de la fonctionnalité du point de vue de l'utilisateur

2. Conception visuelle : La densité d'information sur l'écran est-elle raisonnable. ? La distribution des éléments sur l'écran est-elle en accorde avec la séquence naturelle des actions ?

3. Efficience : Les clients peuvent-ils compléter les tâches de base dans de laps de temps raisonnables ? Le rythme de travail est-il adéquat ?

4. Simplicité d'apprentissage : L'interface minimise-t-elle la nécessité de connaître la signification des abréviations et des acronymes spécifiques ? Évite-t-elle la nécessité de savoir un jargon technique spécifique ? Le logiciel peut-il être utilisé sans la lecture de documentation ?

5. Navigation : Est-il facile de se déplacer à travers le logiciel ? La navigation est-elle intuitive ? Les clients peuvent-ils trouver facilement l'information qu'ils cherchent ?

6. Consistance : Le logiciel utilise-t-il une terminologie et des conventions consistantes avec l'utilisation générale de ce type de produit logiciel ? Utilise-t-il des icônes standard ?

7. Rétroaction : Le logiciel fournit-il une rétroaction convenable et utile à l'utilisateur ? Les messages d'erreur sont-ils présentés dans un langage simple et précis ? Présentent-ils des instructions claires et dans un ton instructif et positif ? Est-il clair pour l'utilisateur ce qu'il faut faire ensuite ?

8. Accessibilité : Le logiciel est-il conçu pour s'adapter aux utilisateurs ayant des incapacités ?

3.3 Évaluation manuelle vs automatique

Les caractéristiques principales à comparer entre la façon traditionnelle (c'est-à-dire, manuelle) et les différents degrés d'automatisation possibles sont le temps d'évaluation demandé, le budget disponible, l'exigence de professionnels ou d'experts demandée, l'échantillon considéré pour la validation des résultats, la portée de l'évaluation, l'environnement où l'évaluation se fait, la dimension de l'étude réalisée et les types de systèmes à évaluer plus appropriés.

Les éléments dont il faut tenir compte sont présentés dans le tableau 3.2.

3.4 Standards de qualité d'utilisabilité

La norme ISO9126, figure 3.1, constitue un standard international pour l'évaluation de logiciel. Elle définit un modèle de qualité pour un produit logiciel. Le modèle de qualité est divisé en trois sujets : qualité interne, qualité externe et qualité d'utilisation qui conduisent aux métriques externes, métriques internes et métriques de qualité d'utilisation. En ce qui concerne la qualité d'utilisation, la norme définit ce concept à partir de trois caractéristiques mesurables affichées dans le tableau 3.3.

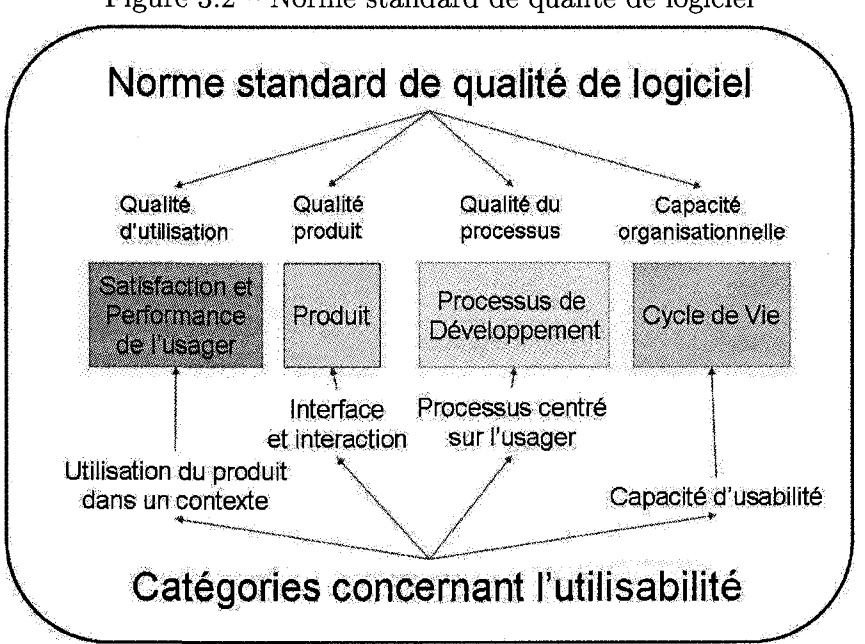
Figure 3.1 – Standard ISO9126

Il existe beaucoup de normes en rapport à l'interaction homme-machine et l'utilisabilité. Elles sont classifiées selon :

1. L'utilisation du produit (efficacité, efficience et satisfaction dans un contexte particulier d'utilisation).
2. L'interaction et l'interface d'utilisateur.
3. Le processus suivi pour développer le produit.
4. La capacité d'une organisation pour appliquer le design centré sur l'utilisateur.

La figure 3.2 présente le rapport entre les différents composants qui contribuent à la qualité de logiciel.

Figure 3.2 – Norme standard de qualité de logiciel



3.5 Étapes du projet

Le projet s'est divisé en cinq étapes :

1. Examiner les documents de la littérature concernant les métriques d'utilisabilité, les techniques et les procédures d'évaluation d'utilisabilité.
2. Définir un sous-ensemble de techniques et de métriques applicables aux logiciels qui appartiennent au type d'application de travail de bureau administratif. Le coût des procédures doit être bas et la réalisation doit prendre peu de temps.
3. Valider ce sous-ensemble de techniques et construire des outils permettant de faciliter l'évaluation d'utilisabilité aussi bien que l'analyse des données recueillies.
4. Évaluer un cas d'utilisation en appliquant les techniques et les outils générés.

Le cas d'utilisation ou cas d'application de la méthodologie d'évaluation d'utilisabilité est le système Diamant composé par les logiciels Diamant (une application autonome) et Diamant Web (une application Web). En lignes générales, la méthodologie d'évaluation d'utilisabilité appliquée dans le cas de ce projet a consisté à une combinaison de techniques. Elle a reposé sur la révision experte du système sous évaluation et sur des entretiens et des

observations directes des utilisateurs. De plus, différents types de questionnaires génériques ont été adaptés aux caractéristiques du logiciel sous évaluation. Dans cette première étape, ces questionnaires ont été élaborés pour l'analyse manuelle. L'automatisation de l'analyse de résultats a été envisagée, à partir d'outils tels que les feuilles de calcul.

Étant donné que le logiciel sous évaluation avait été déjà livré, les techniques post livraison ont été les plus adéquates. Cependant, comme il n'existait pas une étude préalable en ce qui concerne l'analyse de contexte, de profil des utilisateurs et de profil de tâches d'utilisateur, il a été nécessaire de commencer par ces études en tant que préliminaires.

3.6 Aspects généraux et spécifiques

La partie générique de ce projet qui s'applique à n'importe quel type de logiciel comprend la méthodologie appliquée, la formalisation de la description de différentes procédures et la construction des questionnaires et des enquêtes qui sont indépendantes du profil des tâches et des utilisateurs du logiciel sous évaluation. Cette partie se situe dans le cadre d'une méthodologie légère quant au coût et au temps.

La partie spécifique qui s'applique au logiciel sous évaluation comprend les documents, les outils, les questionnaires et les enquêtes qui contiennent de données dépendantes du profil des utilisateurs et des tâches d'utilisateur concernés.

Tableau 3.1 – Métriques d'utilisabilité

Item	Métrique	Description	Technique de mesure
1	Période de réalisation d'une tâche.	Temps employé par l'utilisateur pour réaliser une tâche.	Élaboration de séquences de tâches et mesure de temps de réalisation.
2a	Taux d'erreurs.	Nombre d'erreurs commises par l'utilisateur divisé par le nombre d'actions totales faites (correctes et incorrectes).	Élaboration de séquences de tâches, énumération des actions pour chaque tâche et comptabilisation des erreurs (clic sur un item de menu incorrect, invocation d'annuler et/ou refaire, clic sur un bouton incorrect, etc.)
2b	Productivité.	Nombre d'actions réalisées correctement par l'utilisateur divisé par le nombre d'actions totales faites (correctes et incorrectes).	Élaboration de séquences de tâches, énumération des actions pour chaque tâche et comptabilisation des actions correctes.
3	Période d'apprentissage.	Temps employé pour apprendre comment réaliser un certain nombre de tâches qui représentent la fonctionnalité de base du produit. Période d'entraînement nécessaire.	Mesure du temps jusqu'à ce que la productivité atteinte une certaine valeur ou le taux d'erreur soit plus bas qu'un certain seuil.
4	Capacité de rétention.	Fait de garder quelque chose en mémoire.	Élaboration de séquence de tâches, énumération des actions jusqu'à complétion pour chaque tâche et comptabilisation des accès à l'aide en ligne ou aux manuels. Fréquence d'utilisation des aides ou consultation du support par rapport à la fréquence d'utilisation du produit.
5	Satisfaction subjective.	Opinion de l'utilisateur.	Utilisation de questionnaires.

Tableau 3.2 – Évaluation d'utilisabilité manuelle et automatique

Item	**Évaluation manuelle**	**Évaluation automatique**
1 Temps.	Chronophage (time consuming).	Rapide.
2 Budget.	Coût important (expensive money).	Bon marché, économique.
3 Exigence professionnelle.	Exige des ingénieurs en utilisabilité sur appel.	Automatique, distant.
4 Échantillon.	Petit nombre de personnes pour effectuer les tests.	Grand nombre de personnes pour effectuer les tests.
5 Portée.	Utilisateurs locaux.	Utilisateurs internationaux, inter villes, interrégionaux.
6 Environnement.	Laboratoire contrôlé.	Chez soi, bureau.
7 Dimension étudiée.	Profondeur	Largeur, étendue.
8 Applications.	Tous types de logiciel.	Site Web, E-Commerce, Interfaces WWW.

Tableau 3.3 – Qualité d'Utilisabilité Norme ISO9126

Item	**Caractéristique**	**Characteristic**	**Description**
1	Intelligibilité.	Understandability.	Attribut du logiciel qui porte sur l'effort de l'utilisateur pour reconnaître le concept logique sous-jacent et son applicabilité.
2	Simplicité d'apprentissage.	Learnability.	Attribut du logiciel qui porte sur l'effort de l'utilisateur pour apprendre son application.
3	Opérativité.	Operationability.	Attribut du logiciel qui porte sur l'effort de l'utilisateur pour l'opérer et contrôler son opération.

CHAPITRE 4
MÉTHODOLOGIES D'ÉVALUATION D'UTILISABILITÉ

Le but de ce chapitre est de présenter les principales techniques d'évaluation d'utilisabilité ainsi qu'une étude comparative de leur application. La littérature montre un grand nombre de techniques et de variantes de ces techniques. La description a été faite de façon à éliminer les petites différences qui se trouvent dans les détails de chaque technique.

4.1 Types d'évaluation et de méthodes

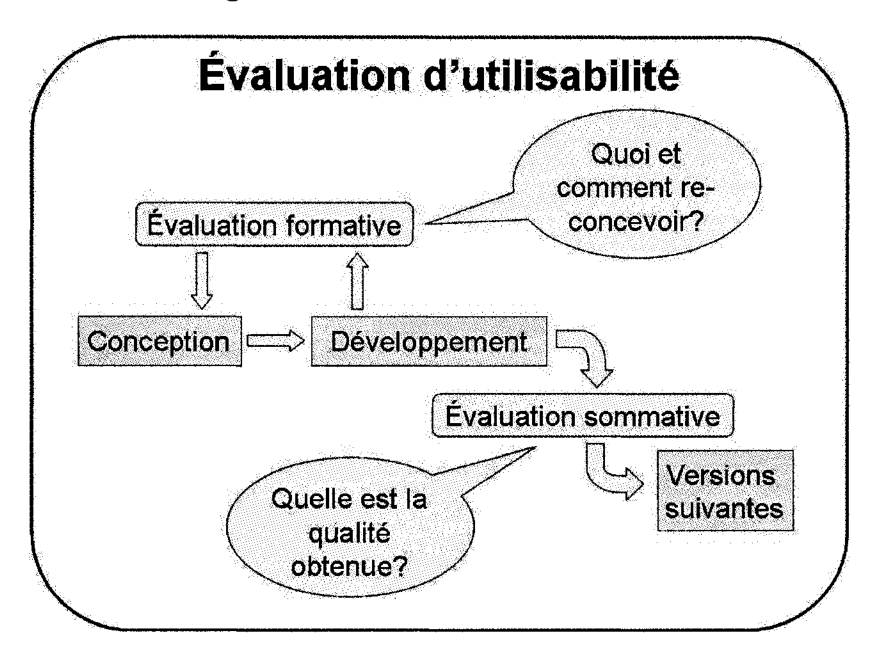
Une évaluation d'utilisabilité peut avoir lieu pendant le processus de développement ou une fois fini. Dans le premier cas, l'évaluation a un caractère formatif où les prototypes et les différentes versions du produit sont évalués. Le but est d'identifier les aspects de la conception que peuvent être améliorés, établir des priorités et fournir une guide de comment faire des changements dans la conception. Par contre, l'évaluation à la fin du développement, de caractère sommatif, vise à la mesure de la qualité, ce qui est fait pour estimer le résultat de la conception.

En ce qui concerne les méthodes d'évaluation, il existe deux types :

1. Les méthodes analytiques
2. Les méthodes empiriques, qui se divisent en :
   (a) Les méthodes empiriques informelles
   (b) Les méthodes empiriques formelles

Les méthodes analytiques sont centrées dans les caractéristiques du système sous évaluation, basées sur un modèle mental de l'utilisateur et sur des guides et des principes de conception d'interaction de l'interface utilisateur. Leur principale faiblesse est la production de plusieurs interprétations possibles qui doivent être validées empiriquement.

Figure 4.1 – Évaluation d'utilisabilité



Les méthodes empiriques impliquent la participation de l'utilisateur réel et peuvent être plus ou moins formelles. Les méthodes empiriques informelles sont basées sur l'observation de l'utilisateur pendant qu'il explore ou utilise le logiciel sous évaluation. Les méthodes empiriques plus formelles incluent des essais dans des laboratoires d'utilisabilité spécialement conçus pour mesurer la performance et les erreurs des utilisateurs. Les données collectées nécessitent une bonne compréhension du système évalué pour être interprétées correctement.

Le tableau 4.1 montre un résumé des méthodes.

En ce qui concerne le présent projet, le logiciel a été déjà livré. Par conséquence l'évaluation est de caractère sommatif et la stratégie adoptée a été :

1. Une première étape d'application d'une méthodologie analytique : une combinaison des méthodes de révision experte, visite experte et évaluation heuristique du produit
2. Une deuxième étape d'application d'une méthodologie empirique d'observation des utilisateurs réels : une combinaison des méthodes "penser en haut" et enquête.

Cette approche permet de proposer une nouvelle stratégie de méthodologie d'évaluation sommative de produits logiciels légère car le temps et le coût doivent être bas.

4.2 Spécifications pour une évaluation d'utilisabilité

L'évaluation commence par la détermination des spécifications constituées par des caractéristiques d'utilisabilité requises du système. Les spécifications reposent sur une analyse des tâches de l'utilisateur. Les tâches sont classées selon la fréquence d'utilisation et l'impact sur le travail global de l'utilisateur. Les tâches sont divisées en sous tâches pour fournir des spécifications plus précises par rapport au besoin de l'utilisateur. Un modèle de tâches et sous-tâches et des spécifications d'utilisabilité pour le logiciel ciblé sont donnés dans l'annexe C.

Les caractéristiques de conception de l'interaction de l'interface d'utilisateur analysées pour le logiciel ciblé sont décrites dans le chapitre 6 et l'application pratique est présentée dans l'annexe B.

4.3 Méthodes analytiques

4.3.1 L'analyse de demandes

L'analyse de demandes est une méthode où un ingénieur en utilisabilité identifie les caractéristiques importantes et génère des hypothèses par rapport aux conséquences que ces caractéristiques pourraient avoir sur des utilisateurs engagés dans des tâches (activités, scénarios, etc.). Une demande est un énoncé sur un certain aspect de l'interaction qui a une influence psychologique sur l'utilisateur et qui se reflète dans la façon d'interagir de ce dernier. Des exemples des aspects tenus en compte peuvent être : le placement d'un bouton sur une fenêtre, le style d'une barre de défilement, la retroaction fournie comme résultat de l'invocation d'une commande par l'utilisateur, etc. Les composants de l'interface d'utilisateur sont listés avec ces caractéristiques et avec les implications positives et négatives sur l'utilisateur. Cette approche permet de choisir parmi un ensemble de possibilités de conception et de clarifier des questions à analyser pendant une évaluation avec l'utilisateur. Un avantage est qu'il s'agit d'une technique "avant et pendant" le développement, c'est-à-dire qu'elle peut s'appliquer avant la construction de prototypes et avant les évaluations avec les utilisateurs. Un désavantage est que le concepteur peut se sentir évalué : il essaie de développer le meilleur produit et les problèmes d'utilisabilité rapportés peuvent être ressentis comme des critiques. Il est nécessaire de promouvoir une ambiance coopérative entre les

concepteurs et les ingénieurs d'utilisabilité afin de raffiner la conception du produit.

4.3.2 L'inspection d'utilisabilité

Les guides d'utilisabilité constituent un outil traditionnel du processus de développement de logiciel. Il existe dans la littérature plusieurs références, listes, exemples et exceptions qui conforment les documents de guide. Les guides sont la base des inspections d'utilisabilité sur lesquelles des experts et non experts en utilisabilité examinent et/ou essaient de travailler avec le système sous évaluation dans le but de détecter des problèmes potentiels. Les documents de guide d'utilisabilité sont des outils d'évaluation plus que de conception. Sa nature ouverte et incomplète ne permet pas de généraliser. Ils ne peuvent pas être considérés pour toutes les caractéristiques possibles de design et tous les problèmes potentiels. Il faut extraire, adapter et retenir certaines parties de ces guides adéquats pour l'application sous évaluation. Il n'existe pas des lignes ou des guides absolument générales.

Les méthodes d'inspection peuvent se classer selon les catégories montrées dans le tableau 4.2. Essentiellement, ils consistent en des révisions faites par des experts ou des personnes avec une forte expérience dans le domaine et/ou de l'application. Les révisions peuvent comprendre une critique de l'interface d'interaction, un parcours à travers des tâches en simulant l'utilisateur et/ou la vérification de la conformité à certaines heuristiques. La technique de parcours consiste à une révision pas à pas des spécifications et des caractéristiques d'utilisabilité par le biais de la réalisation des tâches d'utilisateur.

En termes généraux, les méthodes d'inspection d'utilisabilité permettent de mettre en évidence un grand nombre de problèmes potentiels d'utilisabilité à un coût raisonnable. Cependant, l'inspection n'est pas suffissante pour valider les constatations comme des problèmes réels d'utilisation du système

Ces méthodes visent à trouver et comptabiliser des problèmes plus qu'à comprendre les causes et les effets d'un problème. Elles fournissent une rétroaction rapide. Par contre, elles peuvent attirer l'attention sur des problèmes rares, atypiques ou négligeables dans l'utilisation réelle. Pour cette raison, il est convenable d'inclure dans le rapport de l'inspection une classification des problèmes selon une échelle de sévérité par rapport à l'effet sur l'utilisateur.

L'une des questions à propos de ces techniques concerne la contribution qu'elles apportent à la discipline de l'ingénierie en utilisabilité. En effet, bien qu'elles soient des

méthodologies légères, faciles à mettre en œuvre, il est difficile de conserver et de réutiliser les idées et connaissances développées par un évaluateur dans un processus d'évaluation afin de leur utilisation dans d'autres projets par d'autres évaluateurs.

4.3.3 Analyses basées sur le modèle mental de l'utilisateur

Le but de ce type d'analyse est d'établir des théories afin de construire un modèle prédictif. La représentation de la connaissance et des buts de l'utilisateur constitue un modèle d'utilisateur. Par le biais de ces modèles, il est possible d'étudier la complexité et la consistance des interfaces et la performance de l'utilisateur. Ces techniques peuvent produire des prédictions précises par rapport au comportement de l'utilisateur à bas niveau, mais elles prennent peu en compte les aspects du comportement à haut niveau tels que la capacité d'apprentissage, de résolution de problèmes et les interrelations sociales des utilisateurs. Les éléments considérés comprennent

1. La vitesse d'accomplissement de micro tâches : faire une entrée au clavier, pointer avec la souris, cliquer sur un bouton, réaliser une opération mentale (penser)
2. L'analyse de l'exécution séquentielle et parallèle de tâches
3. L'utilisation de métaphores qui favorisent l'apprentissage par analogie avec des situations connues, ce qui peut aider à la facilité et à la vitesse d'apprentissage
4. L'association entre la connaissance de la tâche de l'utilisateur et les objets et les actions proposés par l'interaction

Les prédictions fournies par ces techniques permettent de discriminer des tâches plus faciles à exécuter, plus faciles à apprendre et les erreurs qui peuvent arriver. L'application pratique de ces modèles se trouve dans des applications de certain type d'interaction homme-machine dans lesquelles le nombre les frappes de clavier est un facteur important (par exemple, stations de travail d'opérateur téléphonique).

4.4 Méthodes empiriques

Les éléments déterminants d'une évaluation d'utilisabilité demeurent sur des données empiriques. Pour cette raison, les évaluateurs d'utilisabilité ont besoin de connaître ce qui arrive quand les gens utilisent le système dans des situations réelles. Les méthodes analytiques

fournissent des résultats qui suggèrent une ligne d'action de corrections et améliorations, mais ils doivent être validés par les données issues de l'environnement réel d'utilisation.

La validation d'une évaluation empirique n'est pas facile. D'une part, attendre l'installation du système terminé dans l'endroit de travail de l'utilisateur pour étudier l'utilisabilité augmente considérablement la chance d'identifier des problèmes sérieux trop tard et par conséquence, les rend difficiles à résoudre. Trouver des problèmes importants durant cette étape signifie recommencer une partie importante du développement. D'autre part, il est raisonnable de se demander jusqu'à quel point les évaluations faites sur un prototype incomplet dans le laboratoire sont valides en regard à l'utilisation dans l'environnement réel.

Rarement un résultat empirique vise à une conclusion unique et évidente. Ce qui est confus pour un utilisateur dans une situation, ne l'est pas pour un autre, ou pour le même utilisateur dans une autre situation. De plus, les résultats statistiques dépendent largement du nombre et des caractéristiques des utilisateurs étudiés.

4.4.1 Études de terrain

Les études de terrain analysent les activités normales dans l'environnement normal de travail de l'utilisateur, circonstance qu'assure la validité de l'évaluation. Toutefois, elles sont qualitatives, étendues, difficiles à résumer et à interpréter. Pour éviter la dispersion provoqué par un grand nombre de données, les observations devraient s'organiser de façon de focaliser et de catégoriser (ou classer) les problèmes (problèmes de distribution de composants à l'écran, problèmes de retroaction, etc.), ce qui s'appelle analyse de contenu. Les observations sont classifiées selon la sévérité de l'impact sur l'utilisateur ce qui aide à prioriser les améliorations à apporter. Des variantes de ce type d'études sont affichées dans le tableau 4.4.

4.4.2 Test d'utilisabilité

Les études sur le terrain peuvent se faire quand le développement du produit est fini. Pour cette raison, les tests d'utilisabilité dans des laboratoires qui n'ont pas besoin d'installations et/ou de mises à jour du produit dans l'environnement réel, peuvent être utilisés pendant l'étape de développement. De plus, ils peuvent être mieux contrôlés et focalisés, permettant la construction de tâches standardisées.

Pour obtenir des résultats valides, les laboratoires simulent les situations de l'utilisation réelle. Les personnes doivent être représentatives de l'utilisateur réel (âge, habileté, formation, expérience, motivation, etc.). Le péril de ce type de test est de choisir incorrectement des aspects spécifiques où le test se focalise. Cela conduit à des observations non représentatives et ainsi trompeuses.

Le problème de la variabilité dans les résultats des tests est causé par les différences de préférences entre les différents profils d'utilisateurs (débutant, connaisseur, expert). Ce sujet est vraiment un défi pour les études empiriques. Pour extraire des conclusions valides, il faut extraire un patron parmi les résultats exploitant un grand nombre de personnes.

Un laboratoire d'utilisabilité est équipé avec des dispositifs spéciaux pour la capture des données telles que caméras vidéo, fenêtres d'observation avec vitres sans teint, capture d'écran, et logiciel d'enregistrement des actions de l'utilisateur sur l'ordinateur. L'ensemble de tâches est soigneusement construit. Les participants, représentants des utilisateurs, sont placés dans un endroit qui simule l'environnement de travail. Néanmoins, les participants sont isolés des distractions normales, des bruits et de l'interaction avec d'autres employés, circonstances difficiles à reproduire dans le laboratoire.

Les données recueillies permettent une étude de la performance (temps d'exécution de tâches) et l'enregistrement de certaines réactions. Mais la majorité de l'expérience (ce que l'utilisateur pense), est inobservable. C'est pour cette raison qu'une variante suggère l'utilisation du protocole "penser tout haut" qui consiste à inviter l'utilisateur à penser à haute voix pendant la réalisation des tâches. L'utilisateur raconte ses buts, ses plans, ses réactions, ses préoccupations, ses incertitudes, etc. Ces données sont analysées et utilisées pour capturer les difficultés d'utilisation du produit. Même si cette technique fournit des pistes pour découvrir les causes et prévoir les conséquences de problèmes d'utilisabilité, le fait de penser à haute voix n'est pas le comportement usuel des personnes au milieu du travail, et ceci surtout pour les utilisateurs d'ordinateurs. Il arrive que le fait de "penser tout haut" est une tâche additionnelle aux vraies tâches de l'utilisateur. Pour cette raison, les erreurs et la performance sont moins significatives que dans d'autres types d'test. La conclusion est que l'externalisation des buts, des plans et des réactions révèle d'autres sources cognitives inobservables, mais l'autoréflexion peut altérer le comportement de l'utilisateur.

Une autre variante utilisée consiste à demander de manière générale les impressions

subjectives de l'utilisateur par rapport au système complet. Un côté intéressant de la collecte des réactions subjectives est qu'elles peuvent ne pas correspondre avec la performance enregistrée. Par exemple, une fonctionnalité peut améliorer l'efficacité, mais en même temps, elle peut ennuyer à l'utilisateur. Se sentir plus confortable avec un produit logiciel peut ne pas être en accord avec l'efficacité technique du produit.

4.4.3 Expériences contrôlées

Les expériences contrôlées essaient d'aller plus profondément dans l'évaluation d'aspects très spécifiques et au delà des observations de performance et de la satisfaction subjective. Dans ce type d'expériences, un ensemble de variables indépendantes est défini de manière à créer des conditions expérimentales d'intérêt. D'une part, le même profil d'utilisateur est exposé à des variables dont les valeurs sont modifiées pour observer la variation du comportement de l'utilisateur (par exemple différents designs de l'interface d'interaction). D'autre part, une situation fréquente consiste à évaluer le même design avec des utilisateurs de différents degrés d'expertise. Une variable dépendante est le résultat d'une expérience et révèle l'effet de la variation des variables indépendantes. Des paramètres tels que la vitesse pour accomplir une tâche ou un ensemble de tâches, la complexité de la tâche (plus ou moins sous-tâches), le nombre d'erreurs, les réactions subjectives peuvent être des exemples de variables de l'expérience. À chaque variable est associée la définition d'une technique de mesure.

La deuxième composante de ce type de méthode est le choix des participantes qui sont exposées aux différentes expériences. Les personnes recrutées doivent être représentatives de la communauté d'utilisateurs. Un même groupe de participantes peut être exposé à toutes les variations des variables indépendantes ou bien des groupes différents travaillent sur différentes conditions d'évaluation. L'utilisation des mêmes personnes pour de multiples conditions permet de contrôler les effets dus aux différences entre individus. Cependant, l'ordre des expériences peut avoir un effet inattendu. L'assignation des personnes à chaque groupe est aléatoire pour distribuer aléatoirement les déviations provenant de l'âge, la motivation et la formation. Le nombre de participants est un élément important afin d'obtenir des résultats staatiquement valides. L'augmentation de l'échantillon conduit à de meilleurs résultats, bien que dans le cas d'évaluation d'utilisabilité, le coût et l'effort demandé pour une expérience avec un échantillon plus grand puisse ne pas être justifié.

Tableau 4.1 – Méthodes d'évaluation d'utilisabilité

Type de méthode	Définition	Méthodes	Description
Évaluation analytique	Inspection qui implique l'analyse des caractéristiques du système et leur impact sur l'utilisation du produit	Analyse de demandes	Analyse de caractéristiques par rapport aux impacts positifs et négatifs
		Inspection d'utilisabilité	Un ensemble de guides et/ou la connaissance d'un expert sont utilisés comme le point de départ de l'identification et/ou de la prédiction de problèmes d'utilisabilité
		Modèles d'utilisateurs	Une représentation de la structure mentale de l'utilisateur réel et de sa tâche est développée et analysée pour évaluer la complexité, consistance, etc.
Évaluation empirique	Inspection qui implique l'observation et d'autres formes de cueillette de données à partir de l'utilisateur réel	Expériences contrôlées	Une caractéristique ou plus sont manipulés pour étudier l'incidence sur l'utilisation
		Expérience "penser tout haut"	Observation de l'utilisateur qui est invité à penser à haute voix quant à ses buts, ses plans et ses réactions pendant qu'il travaille avec le système
		Étude de terrain	Enquêtes, questionnaires et tout type de retroaction de l'utilisateur sont recueillis à partir de l'environnement réel d'utilisation du produit

Tableau 4.2 – Méthodes d'inspection d'évaluation analytique

Type de méthode	**Définition**	**Synopsis**	**Avantages**	**Désavantages**
Révision experte	Experts en HCI critiquent le produit de façon individuel ou par groupes afin de déterminer les aires qui ont besoin d'être améliorés	Besoin d'experts Pas d'utilisation de guides Pas d'utilisation de scénarios ou tâches	L'expert peut comprendre l'espace complet du problème	Difficultés de généralisation, standardisation ou catégorisation pour appliquer à de nouvelles conceptions
Parcours expert	Experts en HCI utilisent des scénarios ou description de tâches pour se guider dans l'analyse de l'interface	Besoin d'experts Pas d'utilisation de guides Utilisation de scénarios ou tâches	L'expert peut focaliser sur des aires de problèmes connus	Polarisation à cause de la sélection de tâches
Évaluation heuristique	Experts en HCI utilisent un ensemble d'heuristiques (règles empiriques) dans la révision, la catégorisation et la justification des problèmes de l'interface	Besoin d'experts Utilisation de guides courtes Pas d'utilisation de scénarios ou tâches	Rapide à mettre en place Permet de dégager des règles communes	Manque de consensus sur la validité des heuristiques
Parcours heuristique	Experts en HCI utilisent un ensemble d'heuristiques (règles empiriques) et des scénarios de tâches dans l'analyse, la révision, la catégorisation et la justification des problèmes de l'interface	Besoin d'experts Utilisation de guides limités Utilisation de scénarios ou tâches	Il peut focaliser sur des aires de problèmes connus Permet de dégager des règles communes	Polarisation à cause de la sélection de tâches

Tableau 4.3 – Méthodes d'inspection d'évaluation analytique cont.

Type de méthode	**Définition**	**Synopsis**	**Avantages**	**Désavantages**
Guides	Guides de centaines à milles d'items utilisés pour examiner si une interface est conforme à ces règles	Pas besoin d'évaluateurs experts Utilisation d'une longue liste d'items qui représentent des règles ou conseils d'utilisabilité	Bon marché	Lent Parfois un travail fastidieux Pas de consensus sur certaines règles
Parcours de guides	Guides pouvant comporter jusqu'à plusieurs centaines des scénarios de tâches utilisés pour guider l'analyse d'une interface	Pas besoin d'évaluateurs experts Utilisation d'une longue liste d'items qui représentent des règles ou conseils d'utilisabilité Utilisation de scénarios et de tâches	Bon marché La méthode peut focaliser sur des aires de problèmes connus	Lent Parfois un travail fastidieux Pas de consensus sur certains items Polarisation à cause de la sélection de tâches
Parcours cognitif	Scénarios complets de tâches pour simuler le processus cognitif et le modèle mental de l'utilisateur afin de guider l'analyse d'une interface	Utilisation de scénarios et de tâches Accent sur le processus cognitif et la perception de l'utilisateur	Centré sur l'utilisateur La méthode peut focaliser sur des aires de problèmes connus Reconnaissance des buts de l'utilisateur	Le concepteur doit devenir un utilisateur Parfois un travail fastidieux Pas de consensus sur certains items Polarisation à cause de la sélection de tâches Ne comprend pas l'espace complet du problème

Tableau 4.4 – Méthodes d'évaluation sur le terrain

Type de méthode	**Définition**	**Synopsis**	**Avantages**	**Désavantages**
Observations sur le terrain	Observation in situ de l'utilisateur qui réalise des tâches réelles dans l'environnement réel	Besoin d'observateurs experts Analyse contextuelle et ethnographique	Validité des données recueillies	Présence de l'évaluateur peut changer le comportement de l'utilisateur Coûteux Analyse difficile
Entretiens rétrospectifs	L'utilisateur est invité à parler à propos des épisodes bons et mauvais éprouvés avec le produit sous évaluation	Description des épisodes considérés remarquables par l'utilisateur	Validité des données recueillies	Influence de l'interprétation du rapporteur
Incidents critiques	L'utilisateur est invité à rapporter des incidents critiques pendant l'utilisation du produit sous évaluation	Rapport des incidents critiques peu après son occurrence	Validité des données recueillies	Influence de l'interprétation du rapporteur

CHAPITRE 5
MÉTHODOLOGIE DÉVELOPPÉE

5.1 Type d'évaluation

La méthodologie proposée dans ce travail consiste en une évaluation sommative applicable aux logiciels déjà livrés. Le cas d'utilisation (cas d'application de la méthodologie d'évaluation d'utilisabilité développée) est le système Diamant. Ce système est composé par les logiciels Diamant (une application autonome) et Diamant Web (une application Web). Il s'agit d'un système d'aide à la construction d'horaires utilisé par plusieurs facultés de l'Université de Sherbrooke.

À partir d'une analyse contextuelle, certains guides de conception d'interfaces d'utilisateur sont choisis afin de les vérifier. Étant donné que le produit sous évaluation est une application utilisée pour travailler dans un environnement administratif, les guides de conception choisis constituent des éléments généraux pour n'importe quelle application de ce type. Les fonctionnalités plus importantes du logiciel guident le choix d'un ensemble de tâches à essayer de parcourir. Cela est spécifique au logiciel sous évaluation. Ce choix se fait selon une liste de priorités croisées parmi les tâches depuis les plus fréquentes jusqu'aux moins fréquentes et depuis les plus critiques jusqu'aux moins critiques. Un guide de conception d'interfaces est développé dans le chapitre 6.

L'analyse contextuelle comprend l'étude du profil d'utilisateur, des tâches, de l'ensemble utilisateur-tâche et les caractéristiques de l'environnement de développement du logiciel. Cette partie est développée dans le chapitre 6.

5.2 Les participants et le recrutement

En ce qui concerne l'évaluateur, une seule personne avec une assez bonne expérience dans le sujet, avec des connaissances des standards d'utilisabilité et des guides établies, peut concevoir et réaliser toutes les étapes du processus d'évaluation.

Étant donné que les utilisateurs utilisent le logiciel sous évaluation (système Diamant) deux fois tous les quatre mois (c'est-à-dire, six fois par année), le temps disponible pour faire les observations des utilisateurs est très réduit et les moments où pouvoir le faire sont très limités.

Les participants impliqués dans l'évaluation sont des utilisateurs réels du produit dont la collaboration est volontaire. Afin d'avoir un nombre de participants significatifs, il faut attirer leur attention et promouvoir leur collaboration. Une réunion au moins avec eux doit se faire. Le but de cette réunion est de présenter le problème, d'expliquer les caractéristiques de leur intervention et de leur demander formellement leur collaboration.

De courts documents préalables à toute observation ou enquête sont distribués aux utilisateurs dans le même but que la réunion précédente, avec plus de détails.

À partir de cette réunion et des documents, il doit être clair pour l'utilisateur que ce qui est évalué est le produit, plus précisément la qualité en utilisabilité du produit. Au contraire, l'utilisateur peut se sentir lui-même évalué et plus exactement, peut sentir que sa capacité à utiliser le logiciel est jugée.

Les documents générés sont présentés dans l'annexe A. Ces documents sont des exemples guides pour l'évaluation d'autres projets.

5.3 L'évaluation proprement dite

L'évaluation proprement dite, comprend deux étapes :

1. Une phase de diagnostic qui utilise une méthode analytique
2. Une phase de validation qui emploie une méthode empirique

D'abord, l'étape de diagnostic qui consiste à une inspection experte est proposée. L'inspection est construite comme une combinaison de révision experte, parcours expert et parcours heuristique. Ensuite, l'étape de validation des données recueillies est composée par des observations des utilisateurs avec le protocole "penser tout haut" et des études de terrain telles que des enquêtes. L'étape de validation permet aussi de découvrir de nouveaux aspects probablement cachés à l'inspection experte.

Les observations des utilisateurs sont faites par l'évaluateur de manière directe, qui a comme mandat :

1. Observer l'utilisateur pendant qu'il travaille
2. Inviter l'utilisateur à penser à haute voix
3. Prendre note de tout ce qui se passe, sans intervenir, sans aider ou fournir aucun type de support
4. À la fin de l'observation, inviter l'utilisateur à rapporter les principales difficultés qu'il a éprouvés ainsi que les caractéristiques qu'il voudrait garder dans une version future du produit.

La présence de l'évaluateur est un inconvénient parce qu'elle modifie le comportement de l'utilisateur. Le fait de penser à haute voix, constitue un élément anormal dans le travail de l'utilisateur.

La méthodologie et les modèles développés dans ce projet sont résumés dans le tableau 5.1. Le tableau indique :

1. la désignation théorique plus fréquente assignée à la sous-étape
2. une brève description
3. où se trouve sont étude et/ou analyse
4. les outils (présentations, documents, enquêtes, etc.) crées comme exemple dans ce travail

Un résumé graphique des étapes et des éléments principaux de la méthodologie développée est montré à travers les figures qui suivent. La figure 5.1 affiche les étapes et sous étapes de l'évaluation. La figure 5.2 montre les aspects sur lesquels la partie analytique de la méthodologie est basée. La figure 5.3 détaille les variables mesurables d'une évaluation d'utilisabilité. La figure 5.4 affiche les caractéristiques de l'interface d'utilisateur à prendre en compte pendant l'inspection du logiciel. La figure 5.5 montre les éléments fondamentaux de la partie empirique de la méthodologie.

La réalisation des étapes est générale pour n'importe quel type de produit logiciel. L'application de la sous-étape et les résultats obtenus sont particuliers au cas évalué dans ce travail. Ils sont donnés à titre d'exemples possibles de construction et de présentation.

Figure 5.1 – Étapes de la méthodologie développée

Méthodologie

Analyse de profils
- Profil d'utilisateur
- Profil de tâches
- Environnement de développement

Méthode analytique
- Inspection du logiciel
  - Documentation d'utilisateur
  - Interface d'utilisateur

Méthode empirique
- Observation de l'utilisateur
- Rapport d'incidents
- Enquêtes

Figure 5.2 – Caractéristiques de l'inspection du logiciel

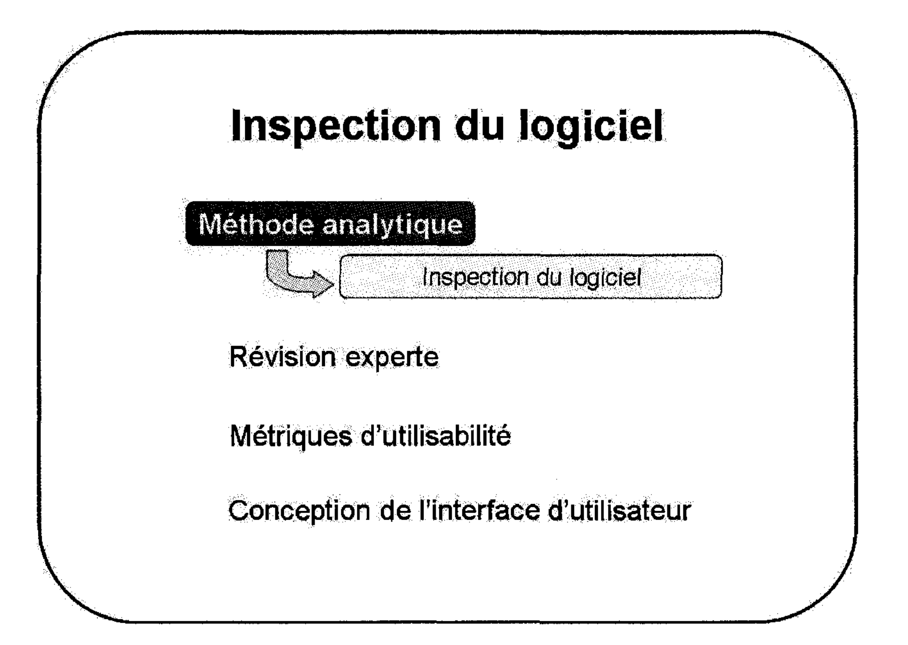


Figure 5.3 – Variables mesurables d'utilisabilité

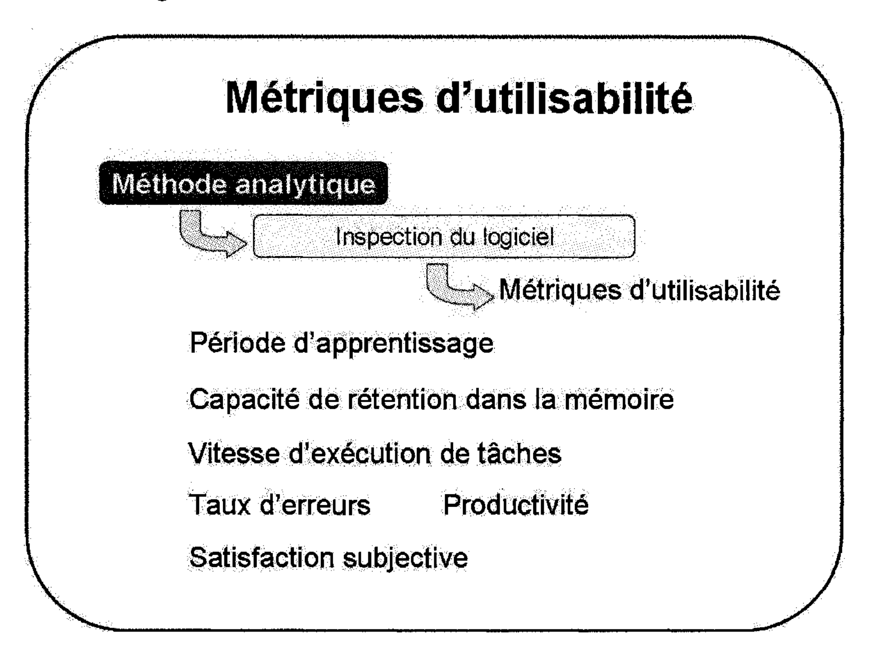


Figure 5.4 – Caractéristiques de l'interface d'utilisateur

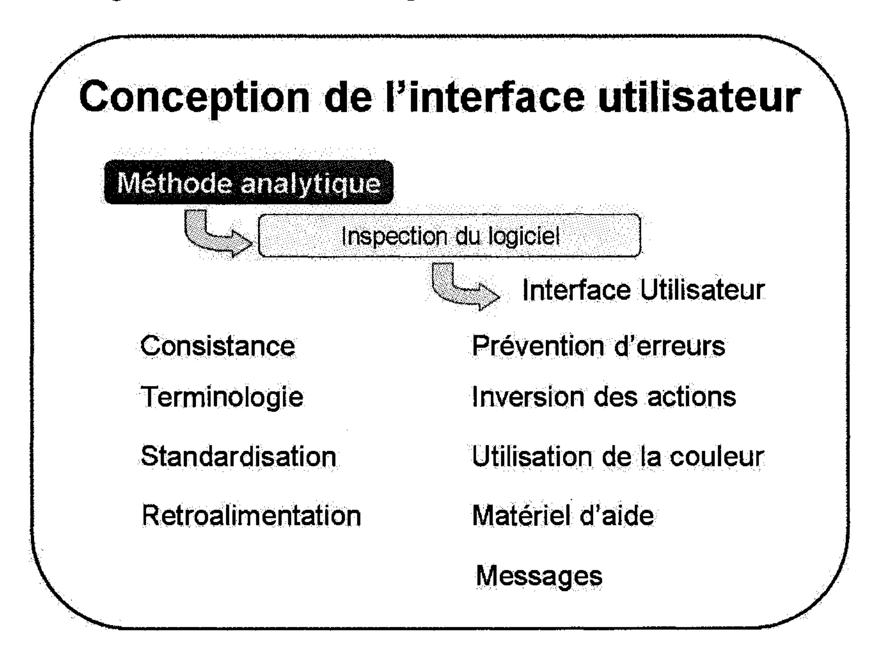


Figure 5.5 – Observation et participation de l'utilisateur

Observation de l'utilisateur

Méthode empirique

Observation de l'utilisateur

Participation de l'utilisateur

1 Observation du travail avec le logiciel sous-test

2 Réponse aux questionnaires spécialement conçus

Tableau 5.1 – Méthodologie développée, étapes et outils

N°	Désignation	Description	Étude et analyse	Documents et outils
1	Analyse contextuelle	Analyse du profil d'utilisateur et des tâches	Chapitre 6	
2	Conception d'interface d'utilisateur	Guide des éléments de conception d'interface d'utilisateur	Chapitre 6	
3	Documents d'intention	Documents qui publient les intentions et la portée de l'évaluation		Annexe A
4	Documents de buts de l'évaluation et de consentement	Documents qui publient les buts de l'évaluation et demandent le consentement des utilisateurs réels pour participer volontairement à l'évaluation		Annexe A
5	Présentation de buts de l'évaluation	Présentation générale à niveau exécutif		Annexe A
6	Méthodes analytiques	Inspections faites par un expert qui impliquent l'analyse des caractéristiques du système et leur impact sur l'utilisation du produit	Aspects généraux développés dans le chapitre 4	
7	Méthodes empiriques	Inspections qui impliquent l'observation et d'autres formes de cueillette de données à partir de l'utilisateur réel	Aspects généraux développés dans le chapitre 4	
8	Enquêtes - Questionnaires	Méthodes empiriques d'étude sur le terrain qui consistent à demander certaines informations et opinions à l'utilisateur		Annexe C
9	Maquette d'interaction	Prototype visuel d'interaction proposé à l'utilisateur pour accomplir une tâche		Annexe C
10	Guide de documentation d'aide	Prototype de guide d'aide orientée sur la tâche de l'utilisateur		Annexe C
11	Résultats des méthodes analytiques	Rapport d'évaluation d'utilisabilité d'un expert		Annexe B Rapport des inspections
12	Résultats des méthodes empiriques	Rapport d'évaluation à partir des observations des utilisateurs		Annexe B Rapport des observations
13	Sommaire exécutif	Rapport destiné aux gestionnaires de l'évaluation d'utilisabilité		Annexe B

CHAPITRE 6
PROFILS ET INTERFACE D'UTILISATEUR

La première partie de ce chapitre présente l'étude de profil réalisée et la deuxième partie détaille les caractéristiques de conception de l'interface d'interaction de l'utilisateur ciblé dans ce travail.

L'étude et l'analyse des profils des utilisateurs et de leurs tâches sont une démarche qui doit se faire avant la conception du logiciel. Une fois le développement du logiciel terminé, si cette étude n'existe pas ou n'est pas documentée, l'évaluation d'utilisabilité doit commencer par établir ces caractéristiques afin de définir l'incidence relative des métriques d'utilisabilité dans l'évaluation. Il s'agit aussi d'une étude du contexte d'utilisation du logiciel sous évaluation.

Une définition des caractéristiques de l'interface d'utilisateur et des conseils afin d'atteindre un haut degré de qualité d'utilisabilité suit l'étude de profil.

6.1 Analyse de profils

6.1.1 Objectif de l'analyse "utilisateur-tâche"

Le premier principe de l'ingénierie d'utilisabilité[10] est "connaître l'utilisateur" ou "know the user". L'idée est simple, mais sous-estimée. Personne n'argumente contre ce principe. Cependant, beaucoup de concepteurs et de développeurs présument qu'ils comprennent bien les utilisateurs et leurs tâches. Le but de cette section est d'établir le profil d'utilisateur et de tâches par rapport au logiciel sous évaluation. Cela signifie d'abord d'identifier les caractéristiques importantes de la communauté d'utilisateurs en ce qui concerne l'interaction avec le logiciel. Ensuite, il faut déterminer les caractéristiques des tâches que les utilisateurs réalisent avec le logiciel ainsi que le degré d'automatisation et les prestations ajoutées au travail que le logiciel fournit aux utilisateurs.

6.1.2 Profil d'utilisateur

Dans le cas du logiciel sous évaluation, l'utilisateur appartient au secteur administratif. Il possède des connaissances minimales de l'ordinateurs en général et de l'utilisation d'un ordinateur pour automatiser des tâches administratives de caractère répétitif, organisationnelles, de planification, d'horaires, etc.

6.1.3 Profil des tâches

Le logiciel sous évaluation est un logiciel d'aide à la construction d'horaires. Il permet de faciliter la tâche de placement temporel et physique des activités telles que cours et examens. Il utilise l'information provenant des activités, des étudiants, des enseignantes et des locaux affectés au différents cours d'une faculté. Le cas de la construction d'horaires est un problème complexe qui n'a pas toujours une solution donc il faut trouver un compromis. C'est pourquoi le logiciel réalise une première assignation des horaires automatiquement et montre tous les conflits présentés. Ensuite il permet de manipuler la préaffectation afin d'éliminer les conflits et de produire l'horaire final.

6.1.4 Profil utilisateur-tâche

L'utilisateur utilise ce logiciel deux fois dans chaque session de quatre mois pendant 8 heures environ à chaque fois, l'une pour générer l'horaire des cours au commencement de la session et l'autre pour assigner les horaires d'examen à la fin de la même session. C'est à dire qu'il s'agit d'un profil d'utilisateur

1. de connaisseur à expert en matière de la tâche administrative à réaliser
2. intermittent, peu fréquent
3. peu connaisseur des concepts associés à la technologie et aux ordinateurs, non expert en matière de la tâche administrative à réaliser

L'utilisateur possède une bonne connaissance sémantique de la tâche qu'il doit réaliser. L'utilisation sporadique du logiciel signifie que l'utilisateur oublie les détails relatifs à comment faire certaines tâches. En général, l'utilisateur a du mal à se rappeler les éléments syntactiques de l'interface, il a besoin d'une interface intuitive en accord avec sa connaissance sémantique, d'une interface où la charge de mémoire à court terme est minimale, de références rapides et d'aide sensible au contexte ou des allusions (hints) pour se souvenir de

comment procéder, de la séquence de tâches et, surtout, des détails. Bref, il sait ce qu'il doit et ce qu'il peut faire, mais il oublie certains détails de comment le faire. Le manque d'expertise sur les concepts orientée-machine (sauvegarder, rechercher, soumettre, copier, remplacer, etc.) ne lui permet pas de trouver rapidement des solutions ou des chemins alternatifs pour faire une tâche quand il a oublié ou partiellement oublié comment le faire.

6.2 Guides de conception de l'interface d'utilisateur

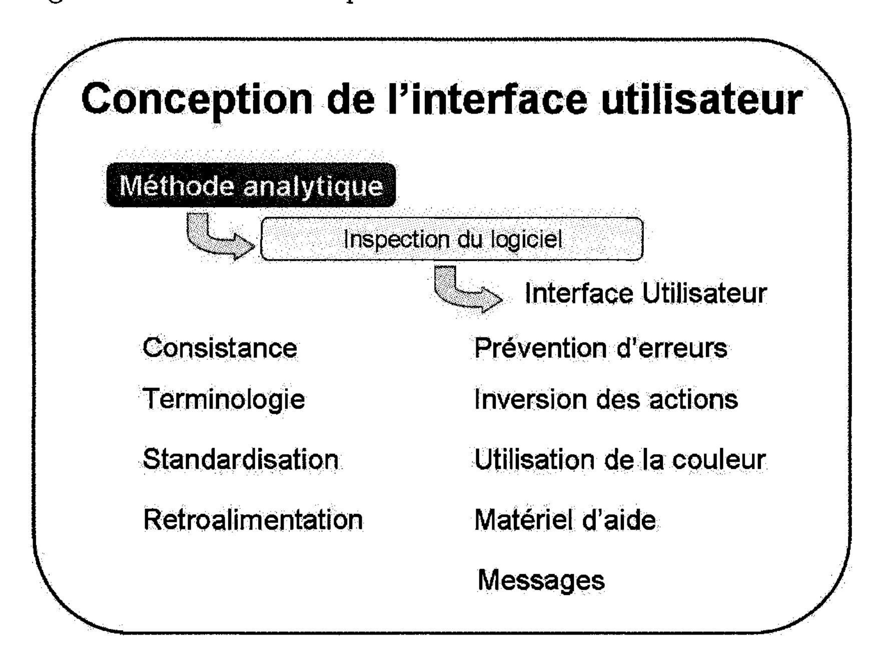
Le but de cette section est d'exposer brièvement les éléments généraux qui caractérisent une évaluation d'utilisabilité[26], en ce qui concerne la conception de l'interface d'utilisateur. De plus, ces éléments constituent les paramètres pris en compte pour évaluer le logiciel. Les mêmes ont été choisis selon les points faibles déjà observés sur le logiciel ciblé. Ils comprennent les caractéristiques suivantes :

1. Consistance de l'interface d'interaction
2. Niveaux de connaissance de l'utilisateur
3. Terminologie employée dans l'interface
4. Standardisation employée dans l'interface
5. Dialogue fermé proposé par l'interface
6. Retroaction fournie par l'interface
7. Matériel d'aide fourni par l'interface
8. Messages fournis par l'interface
9. Prévention d'erreurs fournie par l'interface
10. Possibilités d'inversion des actions fournies par l'interface
11. Utilisation de la couleur dans l'interface

6.2.1 Consistance

La consistance ou cohérence est le caractère d'un système lorsque ses éléments ne sont pas contradictoires. L'harmonie, le rapport logique, l'absence de contradiction sont les caractéristiques trouvées dans l'enchaînement des parties du tout qui constitue un système consistant ou cohérent. Il est possible de parler aussi d'uniformité. Un système dont les parties se tiennent et s'enchaînent avec ordre de manière à former un ensemble logique, harmonieux

Figure 6.1 – Caractéristiques de l'interface d'utilisateur à évaluer



et satisfaisant facilite l'apprentissage et la rétention en mémoire à long terme. L'inconsistance ou incohérence produit de la confusion, est source de mauvaises interprétations, et est génératrice d'erreurs. L'exception est valide seulement lorsqu'il faut attirer l'attention de l'utilisateur. Et bien sûr, il ne faut pas attirer l'attention de l'utilisateur tout le temps, au risque d'obtenir l'effet contraire. La consistance se réfère premièrement aux éléments de l'interaction et de l'interface d'ordinateur tels que :

1. séquences d'actions communes
2. terminologie des menus, des prompts et des aides, mots et abréviations
3. distribution et position des contrôles sur l'écran
4. typographie
5. couleurs
6. formats d'entrée et de sortie

Cependant, la documentation et le matériel d'aide doivent aussi respecter la consistance.

6.2.2 Niveau de connaissance

Le niveau de connaissance de l'utilisateur du point de vue de l'utilisabilité d'un produit logiciel est classé selon :

1. Connaissance syntactique : description des objets et des actions d'interface
2. Connaissance sémantique : séquences d'actions pour accomplir une tâche
3. Connaissance pragmatique : connaissance spécifique du domaine de la tâche

La connaissance sémantique et pragmatique, une fois acquise, reste dans la mémoire à long terme de l'utilisateur. Cette mémoire est permanente et survit malgré un usage peu fréquent. La connaissance sémantique appartient au domaine des tâches et de la signification des procédures. C'est pour cette raison qu'elle est permanente une fois acquise. Par contre, la connaissance syntactique réside dans la mémoire à court terme de l'utilisateur, elle est extrêmement volatile, c'est pourquoi il faut la rafraîchir fréquemment pour la maintenir. La limitation de la capacité de traitement de l'information dans la mémoire à court terme de l'être humain[1] conduit au besoin de réduire la charge de cette mémoire de l'utilisateur afin de faciliter et optimiser l'utilisation d'un produit logiciel, de diminuer la période d'apprentissage et d'améliorer la mémorisation ou rétention. Bien que de manière générale, il soit nécessaire de minimiser tout type de charge de mémoire de l'utilisateur, la charge de mémoire à court terme est la plus critique, spécialement dans le cas d'un utilisateur intermittent.

Les caractéristiques décrites dans cette section appartiennent au domaine de modèles mentaux des utilisateurs.

6.2.3 Terminologie

La terminologie utilisée tout au long de l'interface d'utilisateur doit appartenir au domaine de la tâche. L'ordinateur est un outil de travail pour l'utilisateur et comme tel, il doit être transparent et il ne doit pas ajouter des éléments à apprendre. L'utilisation de la terminologie orientée à la tâche permet de réduire la période d'apprentissage, permet d'améliorer la rétention et augmente la satisfaction subjective. L'ordinateur et le logiciel sont perçus comme un outil d'aide au travail. La terminologie spécifique de la tâche doit être déterminée avec le consentement de l'utilisateur et elle doit être clairement définie pour les nouveaux utilisateurs.

6.2.4 Standardisation

La standardisation aide à l'utilisation. Certains contrôles des interfaces, tels que les boutons, les items de menu, les items de la barre de tâches associés aux fonctions standard

[1] George Miller 1956 Chunk Theory

(fichier, fermer, sortir, aide, etc.) possèdent des caractéristiques, des raccourcis (shortcuts), des touches accélératrices (accelerator keys) standards. Par exemple, la touche F1 est un standard pour invoquer de l'aide. Changer la fonctionnalité de cette touche n'est pas convenable. Satisfaire ces conventions permet de diminuer la période d'apprentissage, d'améliorer la rétention et de travailler plus rapidement. Par contre, le manque de respect de ces conventions entraîne de la confusion et augmente la probabilité d'erreur, à cause du manque de cohérence à travers de différents logiciels.

6.2.5 Dialogue fermé

Les séquences des actions doivent être organisées en groupes avec un début, un milieu et une fin. L'information de rétroaction à l'achèvement d'un groupe d'actions donne le sens d'accomplissement, le signal de préparation pour la séquence suivante d'actions. Un dialogue qui reste ouvert entraîne de la confusion.

6.2.6 Rétroaction

Pour chaque interaction de l'utilisateur, le système doit fournir un signal de rétroaction. Pour des actions atomiques et/ou fréquentes, la réponse peut être limitée tandis que pour des actions inhabituelles et/ou plus importantes la réponse doit être substantielle. Les changements doivent être explicites. De cette façon, le dialogue entre l'homme et la machine est complété et l'utilisateur sait tout le temps ce que la machine a fait ou est en train de faire.

6.2.7 Matériel d'aide

Le matériel d'aide couvre l'écart entre ce que l'utilisateur connaît déjà et ce qu'il a besoin de connaître. L'aide en ligne, l'aide sensible au contexte, les allusions (hints) sont indispensables dans le cas d'utilisateurs peu fréquents, en raison de leur tendance à oublier les détails. Ils ont besoin de rappels rapides qui n'altèrent pas leur rythme de travail et qui ne soient pas distractifs.

6.2.8 Messages

Les messages à l'utilisateur, du point de vue de l'utilisabilité, doivent être constructifs, instructifs s'il est possible et non distractifs. La caractéristique de non distractif dépend

de l'expertise de l'utilisateur. Un utilisateur nouveau a besoin de plus de messages qu'un utilisateur expert. Un utilisateur nouveau, évolue et après une période d'utilisation du logiciel, devient expert. Alors, les messages d'abord nécessaires, deviennent inutiles et distractifs pour le même utilisateur pendant son évolution comme tel. Il faut fournir la possibilité d'éliminer certains messages qui puissent être distractifs pour l'utilisateur expert. Choisir la rédaction appropriée du message, très concise, constructive et instructive est une caractéristique essentielle à l'efficacité du message.

6.2.9 Prévention d'erreurs

La conception doit éliminer la possibilité que l'utilisateur commette une erreur très sérieuse. Il faut éviter que l'utilisateur puisse perdre son travail, peu importe la raison ou le coupable. Si l'utilisateur commet une erreur, l'interface doit la détecter et offrir des instructions précises, simples et constructives pour procéder à la récupération.

6.2.10 Inversion des actions

Autant que possible, les actions devraient être réversibles. C'est-à-dire qu'après une action, le système devrait pouvoir revenir à l'état précédent (undo-redo). Cela donne une sensation de confiance et de sûreté qui habilite l'utilisateur à explorer les fonctionnalités du système sans élever le niveau de stress.

6.2.11 Utilisation de la couleur

La couleur dans une interface d'utilisateur est un élément qui peut tant favoriser que défavoriser l'interaction avec l'utilisateur. D'une part, la couleur aide à mettre en relief la signification de certains éléments d'interface et/ou à l'interprétation d'un état du système quand son pouvoir de codification est correctement utilisé. D'autre part, son utilisation comme détail décoratif doit être limitée. Il est déconseillé de mettre sur l'écran des pairs de couleurs non convenables à la vision. Il faut éviter les couleurs fluorescentes, les pairs trop contrastants ou avec très peu de contraste, c'est-à-dire, respecter en tout cas la lisibilité, la capacité d'absorption d'information et la résolution visuelle.

CHAPITRE 7
RÉSULTATS OBTENUS

Les résultats obtenus comprennent les lignes de base simple pour mettre en œuvre l'utilisabilité dans un projet et des suggestions et de guides afin d'améliorer l'interaction homme-machine présentée par le logiciel sous évaluation. D'après 6.2.2, un but fondamental des suggestions et des guides est de minimiser la charge de mémoire syntactique et à court terme de l'utilisateur. Par contre, il faut augmenter le contenu sémantique de l'interface et s'appuyer sur les concepts qui reposent dans la mémoire à long terme de l'utilisateur. Également, il faut éliminer de cette interaction tous les concepts liés à l'ordinateur, tant du matériel que du logiciel, ainsi que favoriser la terminologie appropriée au domaine de la tâche ou du problème à résoudre par l'utilisateur en utilisant le logiciel. Les suggestions et les guides peuvent appartenir de manière générale aux domaines divers tels que :

1. Séquences de tâches, flux de données et navigation de l'interface
2. Consistance de la terminologie
3. Consistance, respect de la codification et des caractéristiques perceptives de l'utilisation de la couleur
4. Types et caractéristiques de la documentation consacrée à l'utilisateur
5. Types et rédaction des messages d'avertissement et d'erreur
6. Styles d'interaction (ISO 9241 : manipulation directe, langage de commandes, menu, remplissage de formulaires)

7.1 Guide d'améliorations

Les résultats d'une évaluation d'utilisabilité posent un diagnostic et produisent une guide d'améliorations avec des priorités selon les attentes des utilisateurs. La réalisation des changements sur le logiciel reste sous contrôle de l'équipe de concepteurs et/ou de développeurs et parfois, des gestionnaires ou décideurs. Les documents à propos de ce sujet

sont présentés dans les annexes et selon le détail du tableau 5.1 du chapitre 5 . Un résumé de ce qui est présenté dans les annexes est montré dans les sections suivantes.

7.2 La méthode analytique et ses résultats

La méthode analytique appliquée consiste essentiellement à une inspection de logiciel divisée en deux étapes, l'inspection de la documentation destinée à l'utilisateur et ensuite l'inspection de l'interface d'interaction du logiciel. Le résumé des détails est présenté ici.

Méthode	**Analytique : Inspection de la documentation d'utilisateur**
Participants	Un évaluateur
Temps demandé	90 heures
Description	Inspection de la documentation d'utilisateur du logiciel
Travail livré	Détermination des caractéristiques d'utilisabilité à évaluer Rapport : Problèmes trouvés - Solutions - Recommandations Liste détaillée des défauts détectés
Résultats	Manque de consistance Manque de respect des standards d'utilisabilité Emploi d'une terminologie de bas niveau et d'une terminologie spécifique des tâches non définies

Les conseils donnés dans ce cas sont :

1. Établir des règles de consistance selon les standards de qualité.
2. Générer de la documentation d'aide orientée à la tâche et décrire toutes les étapes du début à la fin de la tâche.

Méthode	**Analytique : Inspection du logiciel**
Participants	Un évaluateur
Temps demandé	180 heures
Description	Inspection de l'interface d'interaction du logiciel par le biais de la réalisation des tâches d'utilisateur
Travail livré	Détermination des caractéristiques d'utilisabilité à évaluer Rapport : Problèmes trouvés - Solutions - Recommandations Liste détaillée des défauts détectés
Résultats	Charge de mémoire à court terme élevée Contenu syntactique élevé Manque de consistance

Les conseils donnés dans ce cas sont :

1. Il faudrait éliminer toute nécessité de connaissance de bas niveau. Les utilisateurs ne devraient pas avoir besoin de connaître la structure de bas niveau des fichiers nécessaires à la construction des horaires. Cela implique la création d'une application avec une interface adaptée au profil de l'utilisateur ou bien, un module de vérification et de validation des données. Ce module devrait rapporter les erreurs trouvées et/ou essayer de les corriger et/ou guider l'utilisateur afin de régler la situation.

2. Une aide en ligne, plus qu'un atout, est un impératif étant donné le profil de fréquentation de l'utilisateur. Cette aide doit comporter, au moins, une aide sensible au contexte et allusions (hints). Étant donné le profil d'utilisateur, un rappel des éléments codifiés (abréviations, couleurs, etc.) sous la forme d'aide sensible au contexte et d'allusions où de petites étiquettes (labels) l'aideraient à se souvenir des aspects partiellement oubliés. Par contre, l'aide proprement dite, organisée et complète s'adresse à un utilisateur nouveau. La conception d'une aide en ligne devrait être réalisée de façon à remplir deux besoins : un accès rapide aux détails syntactiques et aux éléments partiellement oubliés à cause du manque de fréquentation du logiciel et une explication plus détaillée, mais conçue par fragments et liens qui permettent le dosage de l'information nécessaire pour le nouvel utilisateur.

3. La fonctionnalité "défaire" (undo) devrait s'implémenter au moins dans le cas de modifications des items de menu "Affectation". De cette manière, il est possible d'essayer des changements, d'observer les résultats par rapport aux conflits et de décider en conséquence.

4. Par rapport aux couleurs, le système sous évaluation utilise au moins 10 couleurs dans les formes et les caractères utilisés, ce qui peut être trop du point de vue de la signification comme élément de codification. D'ailleurs, la codification naturelle n'est pas respectée (rouge indique danger, interdiction - jaune indique avertissement, faire attention, être alerte - vert indique condition normale, permission, etc.). En tout cas, la signification devrait être précisément définie et indiquée explicitement de manière simple, non distractive, sans consommation appréciable d'espace sur l'écran.

5. Le style d'interaction prédominant dans l'interface est la sélection par menu et l'utilisation de contrôles (spin edits, list boxes, etc.) qui préviennent l'erreur d'utilisateur. Cependant, dans la construction de la grille et de l'horaire et dans la tâche d'élimination

de conflits, il faudrait penser en l'utilisation du style de manipulation directe[1], afin d'essayer le replacement des activités plus agile. Il faudrait penser à d'autres façons de représenter graphiquement l'information afin de permettre d'identifier plus rapidement les sources de conflits et la prise de décisions dans le but de les éliminer.

7.3 Défauts relevés

L'inspection détaillée des logiciels a permis de relever 124 items qui comportent des problèmes d'utilisabilité. Pour évaluer la dimension relative que cette valeur signifie, le nombre de fenêtres présentées par le logiciel et de fonctionnalités fournies ont été comptabilisés. Les résultats sont affichés dans le tableau 7.1.

Tableau 7.1 – Relation entre le nombre de défauts relevés et les caractéristiques des logiciels

Logiciel	**Items relevées (1)**	**Fenêtres (2)**	**Fonctions (3)**	**Fonctions utilisées (4)**	**Relation (1)/(2)**	**Relation (4)/(3)**
Diamant	51	28	7	4	1.82	0.57
Diamant Web	73	16	3	1	4.56	0.33
Système	124	44	10	5	2.81	0.50

L'interprétation du tableau 7.1 permet de conclure qu'il y a presque 3 défauts par fenêtre et que le taux de réalisation de tâches est de 50 pour cent environ.

En ce qui concerne le nombre de fenêtres qui son présentées à l'utilisateur et identifiées par lui comme différentes, les valeurs antérieures ont été obtenues selon les tableaux 7.2 et 7.3.

Par rapport aux fonctionnalités, le nombre a été comptabilisé à partir du témoignage des utilisateurs. Les résultats sont affiches dans le tableau 7.4.

[1]Représentation des objets et des tâches de façon graphique et action directe sur eux avec des dispositifs pointeurs (pointing devices).

7.4 La méthode empirique et ses résultats

La méthode empirique appliquée consiste essentiellement aux observations directes et indirectes de l'utilisateur et a la construction des enquêtes pour l'évaluation future des nouvelles versions du logiciel. Le résumé de détails se trouve ensuite.

Méthode	**Empirique : Observation directe des utilisateurs**
Participants	Un observateur - Trois utilisateurs
Temps demandé	4 heures d'observations proprement dites + 8 heures de préparation 90 heures de travail livré
Description	Observation directe et protocole "penser en haut" pour les tâches plus importantes Enquête sur incidents critiques
Travail livré	Rapport du détail des observations Rapport : Problèmes trouvés - Solutions - Recommandations Exemple de maquette d'interface pour la construction de grilles
Résultats	Validation des inspections Validation des problèmes plus critiques L'étude a trouvé que les problèmes plus critiques sont dus surtout à l'interface du logiciel Diamant Web

La recherche révèle que les principales difficultés des utilisateurs face aux logiciels cités proviennent d'un traitement du format des données d'entrée incorrect de la part des logiciels. Cette circonstance est défavorisée par l'utilisation intermittente des logiciels, très peu de temps dans chaque session (trois fois par année) ce qui ne permet pas aux utilisateurs de se souvenir de plusieurs détails afin de surmonter les blocages. De plus, la documentation qui décrit les fonctions des logiciels ne permet pas aux utilisateurs de reconstruire les pas des tâches ou les pas qui ont été partiellement oubliés. Des éléments conflictuels secondaires se trouvent dans la gestion de grilles depuis Diamant Web.

La solution des problèmes posés consiste à :

1. Ajouter un module dont la fonctionnalité est de vérifier et convertir les données d'entrée automatiquement.
2. Re-concevoir l'interface d'interaction de la gestion de grilles.
3. Concevoir une documentation d'aide et une aide en ligne tous les deux orientées aux tâches de l'utilisateur.

Les recommandations à court terme consistent à :

1. Inclure une fonction qui permet de faire dans une seule étape et sans intervention de l'utilisateur, la vérification des données d'entrée et la conversion adéquate depuis Diamant Web à Diamant. Cette fonction, en cas d'erreur détectée en entrée doit fournir au moins des messages explicatifs aux utilisateurs afin qu'ils puissent réviser et corriger les données avant de continuer leur travail.
2. Fournir de la documentation d'aide orientée vers la tâche des utilisateurs qui décrivent pas à pas les tâches les plus fréquentes du début à la fin, avec une terminologie appropriée à la tâche afin de rappeler rapidement les détails partiellement oubliés.

Recommandations à long terme consistent à :

1. Redéfinir la conception de l'interface d'interaction entre les utilisateurs et le logiciel Diamant Web, surtout dans la création de grilles, la vérification des données d'entrée et leur préparation vis-à-vis leur capture par le logiciel Diamant.
2. Utiliser les techniques orientées vers l'utilisabilité, qui brièvement consistent à évaluer l'interaction dès le début de la conception de l'interface proposée avec les utilisateurs réels.
3. Design d'un aide en ligne orientée aux tâches de l'utilisateur. Étant donné le profil de fréquentation de l'utilisateur, une aide en ligne permet un accès rapide aux éléments partiellement oubliés.

Méthode	**Empirique : Observation indirecte des utilisateurs**
Participants	Un évaluateur - Sept utilisateurs
Temps demandé	90 heures de préparation + 8 heures d'évaluation
Description	Mise en pratique d'un modèle de guide d'aide orientée à la tâche
Travail livré	Guide de tâches les plus fréquentes étape par étape
Résultats	100% des utilisateurs ont trouvé la guide correcte et compréhensible et ont pu faire les tâches guidées correctement

Méthode	**Empirique : Enquête aux utilisateurs**
Participants	Un concepteur de questionnaires
Temps demandé	180 heures
Description	Questionnaires de satisfaction subjective pour les utilisateurs à utiliser après livraison des successives versions améliorées du logiciel
Travail livré	Questionnaire destiné aux utilisateurs version 1
Résultats	Recherche future après avoir implémenté les améliorations suggérées

Méthode	**Empirique : Enquête aux développeurs**
Participants	Un concepteur des questionnaires
Temps demandé	90 heures
Description	Questionnaires aux développeurs pour évaluer les techniques de conception centrée dans l'utilisateur appliquées dans l'équipe
Travail livré	Questionnaire destiné aux développeurs version 1
Résultats	Recherche future avant d'implémenter les améliorations suggérées

7.5 Relation effort-avantages

Faire attention aux facteurs humains dans le développement de logiciel interactif entraîne de meilleurs produits, des utilisateurs satisfaits et fidèles et l'augmentation du profit. Les risques sont essentiellement de type organisationnel. La relation effort avantage de l'application des principes de design centré sur l'utilisateur et des évaluations de l'utilisabilité dès le départ d'un projet dépendent du type de produit. Dans le cas de ce projet les avantages[21] à obtenir sont, du point de vue des utilisateurs :

1. Utilisateurs plus à l'aise, moins frustrés avec le produit
2. Utilisateurs qui réalisent leur travail d'une façon plus agréable
3. Utilisateurs qui atteignent leurs buts de travail plus effectivement et efficacement
4. Utilisateurs qui cultivent la confidence et la crédibilité au produit
5. Utilisateurs qui recommandent le produit, il est possible d'augmenter la portée du produit

Les avantages à obtenir du point de vue des fournisseurs, sont :

1. Réduction de temps et de coûts de développement
2. Réduction de coûts de support
3. Réduction des erreurs d'utilisateur
4. Réduction de coûts de test et d'assurance qualité
5. Réduction de coûts de vente et de cycles de ventes
6. Réduction de temps et de coûts d'entraînement
7. Retour sur investissement

Selon l'opinion des experts[4], le coût dû à l'utilisabilité est 10% du budget de projet ou moins, et les bénéfices peuvent être assez supérieurs à ce chiffre, selon le type de produit et d'entreprise. L'utilisabilité est un facteur qui touche la sphère des affaires, des budgets et des décisions stratégiques. Il n'existe pas un rapport direct entre l'effort de programmation pour changer quelques modules du code du logiciel et l'amélioration produite du point de vue de l'utilisabilité. Les résultats sont classifiés selon l'incidence sur l'utilisateur qui se révèle à partir des évaluations. L'apporte des bénéfices est déterminé selon une pondération des facteurs et une échelle de valeurs pré établie.

Dans ce cas, l'étude d'effort s'est fait sur la base du temps employé pour développer le système sous évaluation et le temps pris par l'évaluation d'utilisabilité, ce qui est affiché dans le tableau 7.5.

7.6 Perspectives futures

Les résultats de l'application des améliorations suggérées par ce projet dépendent de deux facteurs. D'une part, des changements effectifs réalisés sur le logiciel ce qui peut prendre beaucoup plus de temps que la durée de réalisation du projet. D'autre part, une étude d'utilisabilité sert essentiellement en tant qu'outil de diagnostic sur le logiciel, ce qui est validé à partir de l'expérimentation avec l'utilisateur réel. De toute façon, les outils et les documents générés sont utiles pour faire des évaluations d'utilisabilité périodiques et fréquentes en vue des nouvelles versions et nouvelles fonctionnalités d'un produit.

Tableau 7.2 – Fenêtres du logiciel Diamant

N°	**Menu principal**	**Sous menu**	**Nombre de fenêtres**	**Total**
1	Fichier	Horaire cycle/examen	1	
		Grille cycle/examen	1	
		Ouvrir	1	
		Enregistrer	1	
		Définir fichier à importer	1	
		Auto-importer	1	
		Exporter	1	7
2	Affectation	Activités	1	
		Groupes	1	
		Enseignants	1	
		Locaux	1	
		Événements	1	
		Grille partielle	1	
		Affectation manuelle	1	7
3	Modification	Activité	1	
		Importation sélective Enseignants	1	
		Importation sélective Locaux	1	
		Importation sélective Activités	1	
		Importation sélective Étudiants	1	5
		Grille partielle	1	
		Affectation manuelle	1	7
4	Optimisation	Affectation initiale	1	
		Construire l'horaire	1	
		Formation de groupes	1	3
5	Rapports		1	1
6	Préférences	L and F	1	
		Option Conflits	1	
		Affichage Simple	1	
		Affichage Détaillé SH	1	
		Affichage Détaillé SV	1	5
			Nombre Totale de fenêtres	28

Tableau 7.3 – Fenêtres du logiciel Diamant Web

N°	**Menu principal**	**Sous menu**	**Nombre de fenêtres**	**Total**
1	Page d'accueil		1	1
2	Gestion de grilles	Nom	1	
		Modifier	1	
		Afficher	1	
		Copier	1	
		Exporter	1	
		Télécharger	—	
		Importer	1	
		Ajouter	1	7
3	Disponibilité	Ajouter	1	
		Copier	—	
		Effacer	—	
		Importer	—	
		Enseignants Individuel	3	
		Enseignants General	1	
		Locaux	—	5
4	Enseignants		2	2
5	Facultés		1	1

Tableau 7.4 – Utilisation des fonctionnalités du système selon les utilisateurs

N°	Logiciel	Fonction rapportées	Fonctions plus utilisées	Nombre fonctions plus utilisées / Nombre fonctions utilisées
1	Diamant	Construire une grille	x	4/7
2		Construire un horaire	x	
3		Affecter	x	
4		Modifier		
5		Optimiser l'horaire		
6		Obtenir des Rapports	x	
7		Modifier l'affichage		
1	Diamant Web	Projet horaire	x	1/3
2		Construction grille	x	
3		Disponibilité des enseignants	x	

Tableau 7.5 – Estimation du coût de temps du projet

Étape	Item	Valeur
Développement du système	Nombre d'heures par jour	5
	Nombre de jours par mois	20
	Nombre de mois par année	11
	Nombre total d'heures par année	1100
	Nombre de développeurs	4
	Temps demandé en heures	22000
Évaluation d'utilisabilité	Nombre d'évaluateurs	1
	Nombre total d'heures	740
	Coût de temps	3.25%

CONCLUSION

Dans ce document, les différents aspects d'une évaluation d'utilisabilité ont été traités :

- le concept d'utilisabilité et son importance selon plusieurs perspectives
- la méthodologie d'évaluation d'utilisabilité en termes généraux
- la méthodologie développée et les résultats obtenus suite à son application à un système livré

Une fois les erreurs de code d'un produit logiciel éliminées, il est possible que les utilisateurs du logiciel ne puissent pas encore travailler efficacement à cause d'erreurs de conception d'utilisabilité. Dans ce projet, une évaluation sommative d'un système logiciel a été réalisée. Ce type d'évaluation est appliqué à un logiciel déjà livré. Après une analyse contextuelle du logiciel sous évaluation, une combinaison de techniques analytiques et empiriques a été proposée. L'approche utilisée a été basée sur l'analyse de données de terrain et la construction de prototypes d'évaluation d'utilisabilité. Les variables prises en compte comprennent les métriques d'utilisabilité et les caractéristiques mesurables établies par les normes ISO9126 et 9241.

Les conclusions fondamentales tirées de ce travail montrent le bénéfice d'éliminer les éléments syntactiques et la charge de mémoire à court terme de l'utilisateur. Ce type de mémoire de l'être humain est trop volatile, ses contenus sont perdus s'ils ne sont pas utilisés fréquemment. Par contre, il faut mettre l'accent sur la sémantique de l'interface et s'appuyer sur les concepts qui reposent dans la mémoire à long terme de l'utilisateur. Ce type de mémoire de l'utilisateur est permanent. Également, il faut éliminer de cette interaction tous les concepts liés à l'ordinateur, tant matériel que logiciel, ainsi que favoriser la terminologie propre au domaine de la tâche ou du problème à résoudre par l'utilisateur en utilisant le logiciel. Bref, l'ordinateur doit être un outil transparent qui n'ajoute pas d'obstacles au travail de l'utilisateur.

Une intéressante perspective de travail futur consiste à adapter et à compléter les documents et outils crées pour le cas d'une évaluation formative, ce qui signifie l'évaluation d'utilisabilité réalisée dès le départ d'un projet logiciel.

ACRONYMES ET EXPRESSIONS EN ANGLAIS

Sigle	Expression en anglais	Expression en français
	Usability	Utilisabilité , usabilité
	User	Utilisateur, usager
MMI	Man-Machine Interface	Interaction homme-machine
	User Interface	Interface d'utilisateur, interface d'usager
HCI	Human-Computer Interaction	Interaction personne-machine
HCI	Human-Computer Interface	Interface personne-machine
ISO	International Standard Organisation	Organisation des standards internationaux
XP	Extreme Programming	Programmation extrême
	Usability Test	Test d'utilisabilité
	Usability Lab	Laboratoire d'utilisabilité
	Customer Centered	Centré sur le client
	User Centered	Centré sur l'utilisateur
	Scheduling - Timetable	Horaire

ANNEXE A
MODÈLES DE DOCUMENTS

A.1 Introduction

L'objectif de cette section est de présenter les documents qui servent à déclarer les principes ou les intentions sur lesquels l'évaluation d'utilisabilité est faite. D'une part, l'une des étapes de l'évaluation implique la participation d'évaluateurs et d'utilisateurs qui travaillent ensemble : l'utilisateur réalise de tâches prédéfinies et l'évaluateur observe le déroulement des actions de l'utilisateur et prend de notes sans aucune intervention. D'autre part, les résultats des évaluations d'utilisabilité sont adressés aux concepteurs et aux développeurs, qui sont ceux qui vont décider les changements ou les corrections à faire, s'il y a lieu. Tous les participants, mais surtout les utilisateurs doivent savoir que le propos de l'évaluation n'est pas d'évaluer leurs compétences ou leurs connaissances. La cible de l'évaluation est le logiciel et, notamment, ses caractéristiques en ce qui concerne sa facilité d'utilisation de la part des utilisateurs réels. Également, l'objectif de l'évaluation ne porte ni sur la conception du logiciel ni sur les ressources techniques utilisées, même si à partir des suggestions sorties de l'évaluation, des changements dans ce niveau pouvaient être convenables. Finalement, le rôle de l'évaluateur est analysé et ses tâches en termes généraux sont détaillées.

La bibliographie et les recherches à propos de l'évaluation se basent surtout sur des procédures réalisées pendant le développement et avant la livraison du logiciel. Une question majeure dans cette première expérience est que l'évaluation se fait avec des utilisateurs qui déjà connaissent le logiciel, qui ont déjà travaillé avec lui. De plus, ils sont des utilisateurs captifs, c'est-à-dire, qu'ils n'ont pas de choix, ils sont obligés à travailler avec ce produit. Il faudrait construire une deuxième expérience avec des personnes qui n'ont jamais travaillé avec le logiciel. Les documents construits dans cette étape ne subissent pas des modifications pour cette cause.

A.2 Intentions en ce qui concerne les utilisateurs

A.2.1 Objectif

La section suivante a pour but de définir le texte modèle du contenu du document à présenter à l'utilisateur avant de commencer l'évaluation. Le but est de l'informer et de le situer par rapport aux caractéristiques de son intervention dans le processus d'évaluation d'utilisabilité et, bien sûr, d'obtenir son consentement.

A.2.2 Document destiné aux utilisateurs

Le document destiné aux utilisateurs doit contenir au moins, les sections suivantes :

1. Introduction
2. Qu'est-ce qu'une évaluation d'utilisabilité ?
3. Quelles sont les caractéristiques de la participation des utilisateurs ?
4. Conclusion
5. Consentement

Un modèle de document se trouve dans le tableau A.3. Il a été utilisé lors de l'évaluation du système Diamant.

A.3 Intentions en ce qui concerne les développeurs

A.3.1 Objectif

Le texte suivant présente un modèle du contenu du document à présenter aux concepteurs et aux développeurs avant de présenter les résultats de l'évaluation d'utilisabilité. Le but est de les informer et de les situer par rapport aux caractéristiques de son intervention dans le processus d'évaluation d'utilisabilité.

A.3.2 Document destiné aux développeurs

Le document destiné aux concepteurs et aux développeurs doit contenir au moins, les sections suivantes :

1. Introduction
2. Qu'est-ce qu'une évaluation d'utilisabilité ?
3. Conclusion

Un modèle de document se trouve dans le tableau A.5. Il a été utilisé lors de l'évaluation du système Diamant.

A.4 Intentions en ce qui concerne les évaluateurs

A.4.1 Objectif

Dans cette section nous allons passer revues au rôle de l'évaluateur. L'idée est d'observer sans intervenir et sans modifier l'environnement naturel de l'utilisateur. Cependant, la ligne qui sépare d'avoir un effet distractif ou une influence sur l'utilisateur n'est pas bien définie.

A.4.2 Document destiné aux évaluateurs

Le document destiné aux évaluateurs, dans cette première étape consiste à détailler le rôle et les responsabilités des évaluateurs en ce qui concerne les évaluations faites avec l'utilisateur. Il doit contenir au moins, les sections suivantes :

1. Introduction
2. Qu'est-ce qu'une évaluation d'utilisabilité ?
3. Quelles sont les responsabilités des évaluateurs ?
4. Conclusion

Un modèle de document se trouve dans le tableau A.7. Il a été utilisé lors de l'évaluation du système Diamant.

A.5 Document de planification de la participation des utilisateurs

Cette section a pour objectif de présenter la planification de la participation de l'utilisateur des logiciels Diamant et DiamantWeb.

L'utilisateur d'un produit logiciel peut se classer dans une de trois catégories : nouveau, connaisseur et expert du logiciel sous évaluation. Le premier n'a jamais travaillé avec le logiciel. Le deuxième connaît le logiciel et l'a peu fréquemment utilisé, le troisième possède une connaissance approfondie à partir de l'utilisation fréquente et depuis longtemps du logiciel. Il faudrait tenir compte l'évolution de l'utilisateur, c'est-à-dire qu'un utilisateur nouveau aujourd'hui sera connaisseur et même expert après une période d'utilisation du logiciel raisonnable.

La participation de l'utilisateur qui déjà connaît et qui travail depuis longtemps avec le logiciel consiste à deux étapes :

1. Observation de l'utilisateur par un évaluateur d'utilisabilité
2. Questionnaires à l'utilisateur

Dans la première étape, l'utilisateur est observé pendant l'exécution de tâches de son travail habituel avec le logiciel par un évaluateur d'utilisabilité. L'évaluateur ne doit pas intervenir dans l'interaction "utilisateur-logiciel sous évaluation", il doit seulement écouter et prendre de notes sur les commentaires, critiques, indications, suggestions que l'utilisateur dit pendant l'évaluation. Étant donné que les logiciels sont utilisés 2 fois par session (construction des horaires et des examens) il est nécessaire de cordonner l'observation des tâches le plutôt possible afin de remettre les résultats le plutôt possible aussi. Dans la deuxième étape, l'utilisateur doit répondre à une suite de questions ou d'interrogations écrites pour servir de guide et d'orientation dans l'évaluation d'utilisabilité des logiciels sous évaluation. Un rapport destiné aux utilisateurs sera construit à partir de l'analyse des données prises pendant les deux étapes précédentes.

Tableau A.1 – Étapes d'évaluation d'utilisabilité

N°	Étape	Date estimée
1	Observations	Octobre-Novembre-Décembre 2006
2	Questionnaires : Rapport d'incidents	Novembre-Décembre 2006
3	Présentation de résultats	Février-Mars 2006
4	Questionnaires : Satisfaction subjective	Recherche future après corrections des défauts

En ce qui concerne l'utilisateur qui est nouveau avec le logiciel, il existe deux contextes, le premier dans le cas de l'utilisation de la version actuelle du logiciel et le deuxième dans le cas du développement d'une nouvelle version.

En ce qui concerne l'utilisation de la version actuelle par un utilisateur nouveau, la participation consiste à suivre une période d'entraînement avec le logiciel et après, les mêmes étapes que pour un utilisateur habituel du logiciel.

Par rapport à l'utilisation d'une nouvelle version du logiciel, la participation de l'utilisateur qui ne connaît pas et qui n'a jamais travaillé avec le logiciel consiste à assister à des réunions de définition d'exigences du logiciel et à interagir avec le groupe de développement afin de signaler quelque déviation des exigences observées et de compléter les spécifications requises.

A.6 Présentation de buts aux utilisateurs

Avant de commencer les observations directes du travail de l'utilisateur, il faudrait s'assurer que l'utilisateur comprend les buts de l'évaluation d'utilisabilité. Il est important de souligner que l'objectif est l'évaluation du système logiciel, pas la capacité de l'utilisateur de se servir du système. Dans le cas des utilisateurs captifs, une réunion générale des utilisateurs (dans ce cas-ci à peu près une dizaine des personnes, gestionnaires inclus) a été organisée pour exposer les buts dans une présentation du type Power Point. Le fichier qui contient cette présentation s'appelle PresUsagersOct2006-V03.pdf et il est sur le serveur dans le répertoire correspondant au projet.

Tableau A.2 – Modèle de document d'intention destiné aux utilisateurs

Document d'intention destiné aux utilisateurs

Introduction

Le groupe eXit de la Faculté de Génie de l'Université de Sherbrooke travaille depuis quelques années dans un logiciel de construction d'horaires qui fonctionne dans plusieurs facultés de cette Université. La planification et l'implémentation d'une évaluation d'utilisabilité est l'une des étapes considérées d'importance stratégique pour la continuation et l'amélioration de la production de nouvelles versions du logiciel. L'un des membres de l'équipe a présenté un projet de recherche afin de déterminer les concepts de base, d'explorer les principales techniques d'évaluation et de générer un premier ensemble d'outils afin d'évaluer l'utilisabilité des produits logiciels qui seront appliqués dans le sein du groupe de développement. La méthodologie suivie pour atteindre cet objectif doit respecter certaines contraintes de coût et de temps.

Qu'est-ce qu'une évaluation d'utilisabilité ?

Une évaluation d'utilisabilité est un test visant à établir si le système mis en œuvre remplit ses fonctions premières telles que déterminées par ses utilisateurs. Une évaluation d'utilisabilité essaie de mesurer les facteurs de qualité d'une interface utilisateur-ordinateur (interaction homme-machine) d'un logiciel. Le concept d'utilisabilité peut être vu depuis différentes perspectives. D'une part, représente la qualité d'un matériel ou d'un logiciel qui est facile et agréable à utiliser et à comprendre, même par quelqu'un qui a peu de connaissances en informatique et/ou en ordinateurs. D'autre part, dans le cas des logiciels utilisés pour travailler, l'utilisabilité signifie la transparence du logiciel en tant qu'outil, qui permet à la personne d'avancer dans son travail et d'augmenter sa productivité, sans introduire plus des contraintes ou des obstacles additionnels que les propres à sa tâche. Finalement, bien que souvent soit utilisé avec le sens de "convivialité", le terme utilisabilité possède un sens plus large qui correspond "à la capacité d'un système à permettre à ses utilisateurs de faire efficacement ce pour quoi ils l'utilisent". Afin que le travail soit fait, le système utilisable doit non seulement être facile à utiliser, mais aussi fiable et efficace. L'utilisabilité est l'un des domaines de l'interaction personne-ordinateur. Elle diminue l'écart entre les gens et les machines. L'interface d'utilisateur ou interface humain-ordinateur se réfère à la partie du matériel et/ou du logiciel qui permet à une personne de communiquer avec un ordinateur. Cela inclut les dispositifs d'entrée (la façon de parler de la personne avec la machine) et de sortie (la façon de parler de la machine avec la personne). Chacun de ces composants d'interface possède des dispositifs correspondants aux chaînes visuelles, audio et tactiles du cerveau humain. L'utilisabilité étudie ces éléments de l'expérience de l'utilisateur. Par conséquent, l'un des éléments fondamentaux d'une évaluation d'utilisabilité contemple la participation active de l'utilisateur réel.

Quelles sont les caractéristiques de la participation des utilisateurs dans cette évaluation ?

D'abord, l'utilisateur doit savoir qu'il n'est pas l'objet de l'évaluation, qu'il n'est pas évalué, ce ne sont pas ses compétences, ses connaissances ou sa motivation les éléments à évaluer. La cible de l'évaluation est le logiciel. C'est l'interface du logiciel ce qui est sous étude. La participation de l'utilisateur comprend deux aspects : l'étape d'observation pendant la réalisation de tâches avec le logiciel et l'étape de réponse aux questionnaires. Dans le premier cas, le logiciel est évalué à travers l'observation du travail de l'utilisateur avec le logiciel, la façon de faire les tâches poussées par le logiciel, les problèmes trouvés, les commandes de l'interface oubliées, les erreurs plus fréquentes commises dans la réalisation des tâches, les moments d'incertitude, de doute. L'utilisateur peut faire des commentaires ou exposer à voix haute ce qu'il pense pendant la réalisation des tâches. L'évaluateur ne doit pas intervenir dans l'interaction de l'utilisateur avec la machine, il ne doit pas donner des instructions, il doit observer, écouter et prendre des notes sur des indications pertinentes. Après une période d'une à trois heures, l'utilisateur est encouragé à faire des commentaires, des suggestions en termes généraux et/ou à répondre des questions. L'atmosphère doit être informelle de façon à altérer le moins possible l'environnement de travail familier à l'utilisateur. Dans le deuxième cas, l'utilisateur doit répondre une série de questionnaires spécialement désignés. Les questionnaires comprennent des aspects objectifs de l'interaction ainsi que l'avis personnel de l'utilisateur.

Tableau A.3 – Modèle de document d'intention destiné aux utilisateurs (cont.)

Conclusion Les résultats obtenus à partir de l'évaluation d'utilisabilité du logiciel Diamant vont permettre de décider l'implémentation des changements nécessaires sur le logiciel afin d'améliorer l'interaction de l'utilisateur final du logiciel. La prémisse est que le dernier mot en ce qui concerne le succès d'un produit logiciel appartient à l'utilisateur final et qu'un haut degré de satisfaction subjective de l'utilisateur est la clé de l'évolution de n'importe quel projet logiciel. Voilà la raison pour laquelle nous demandons le consentement de participation dans cette évaluation qui termine ce document.
Consentement Je participe volontairement dans cette expérience. J'ai été informé en avance sur les tâches et les procédures à suivre. J'ai eu l'occasion de demander dont j'avais besoin par rapport à cette expérience et j'ai été satisfait avec les réponses fournies. J'ai le droit de me retirer et de discontinuer ma participation sans aucun préjudice futur. Ma signature confirme l'affirmation des énoncés présentés et il précède ma participation effective dans cette expérience.
Nom-Prénom..**Signature**......................................

Tableau A.4 – Modèle de document d'intention destiné aux développeurs

Document d'intention destiné aux développeurs

Introduction

Le groupe eXit de la Faculté de Génie de l'Université de Sherbrooke travaille depuis quelques années dans un logiciel de construction d'horaires qui fonctionne dans plusieurs facultés de cette Université. La planification et l'implémentation d'une évaluation d'utilisabilité est l'une des étapes considérées d'importance stratégique pour la continuation et l'amélioration de la production de nouvelles versions du logiciel. L'un des membres de l'équipe a présenté un projet de recherche afin de déterminer les concepts de base, d'explorer les principales techniques d'évaluation et de générer un premier ensemble d'outils afin d'évaluer l'utilisabilité des produits logiciels qui seront appliqués dans le sein du groupe de développement. La méthodologie suivie pour atteindre cet objectif doit respecter certaines contraintes de coût et de temps.

Qu'est-ce qu'une évaluation d'utilisabilité ?

Une évaluation d'utilisabilité est un test visant à établir si le système mis en œuvre remplit ses fonctions premières telles que déterminées par ses utilisateurs. Une évaluation d'utilisabilité essaie de mesurer les facteurs de qualité d'une interface utilisateur-ordinateur (interaction homme-machine) d'un logiciel. Le concept d'utilisabilité peut être vu depuis différentes perspectives. D'une part, représente la qualité d'un matériel ou d'un logiciel qui est facile et agréable à utiliser et à comprendre, même par quelqu'un qui a peu de connaissances en informatique et/ou en ordinateurs. D'autre part, dans le cas des logiciels utilisés pour travailler, l'utilisabilité signifie la transparence du logiciel en tant qu'outil, qui permet à la personne d'avancer dans son travail et d'augmenter sa productivité, sans introduire plus des contraintes ou des obstacles additionnels que les propres à sa tâche. Finalement, bien que souvent soit utilisé avec le sens de "convivialité", le terme utilisabilité possède un sens plus large qui correspond "à la capacité d'un système à permettre à ses utilisateurs de faire efficacement ce pour quoi ils l'utilisent". Afin que le travail soit fait, le système utilisable doit non seulement être facile à utiliser, mais aussi fiable et efficace. L'utilisabilité est l'un des domaines de l'interaction personne-ordinateur. Elle diminue l'écart entre les gens et les machines. L'interface d'utilisateur ou interface humain-ordinateur se réfère à la partie du matériel et/ou du logiciel qui permet à une personne de communiquer avec un ordinateur. Cela inclut les dispositifs d'entrée (la façon de parler de la personne avec la machine) et de sortie (la façon de parler de la machine avec la personne). Chacun de ces composants d'interface possède des dispositifs correspondants aux chaînes visuelles, audio et tactiles du cerveau humain. L'utilisabilité étudie ces éléments de l'expérience de l'utilisateur. Par conséquent, l'un des éléments fondamentaux d'une évaluation d'utilisabilité contemple la participation active de l'utilisateur réel.

L'utilisabilité du point de vue des concepteurs et des développeurs

Les concepteurs et les développeurs, face aux problèmes d'utilisabilité, devraient se poser la question suivante : sommes-nous vraiment centrés sur le client ? En effet, la première étape pour survivre à une analyse d'utilisabilité consiste à être sûr que le processus de développement est centré sur l'utilisateur. Cela signifie que les utilisateurs sont impliqués dans le processus de conception et d'évaluation du système. L'évaluation proposée aide à mesurer la situation actuelle et fournit une structure visant à l'amélioration. Il permet d'identifier les points forts et les points faibles dans le processus actuel de développement. L'importance de l'utilisabilité du point de vue du développeur réside dans le fait de signifier la différence entre la réussite ou l'échec d'un système. Il faut être conscient des points suivants et agir en conséquence :

- La façon de faire des utilisateurs est presque impossible à prédire
- Les problèmes qui rencontreront les utilisateurs sont très difficiles à prédire
- Les problèmes de l'utilisateur peuvent être inconcevables du point de vue des concepteurs et des développeurs
- La confiance de l'utilisateur est l'un des éléments plus fragiles dans l'interaction homme-machine

Tableau A.5 – Modèle de document d'intention destiné aux développeurs (cont.)

L'utilisabilité du point de vue des gestionnaires Il est fréquent d'essayer de convaincre les gestionnaires de la nécessité d'investir dans l'utilisabilité en faisant appel à la logique, par le biais de l'analyse coût-avantages. La qualité de ce facteur augmente les revenus, crée des clients fidèles, améliore la valeur de la marque et les résultats du processus interne. Il existe assez d'études et des recherches qui confirment ces assertions. Une pauvre conception d'utilisabilité, du point de vue de la gestion, réduit la productivité des efforts de travail, augmente les délais et définit en grande mesure la réussite ou l'échec d'un système. Cependant, il est aussi fréquent de constater que les arguments sont écoutés et après, sont pratiquement ignorés. La raison principale est que la décision sur l'institutionnalisation de l'utilisabilité dans une organisation ne dépend pas d'un argument logique. La conception et la planification centrée sur l'utilisateur ont un impact sur chaque partie de cette organisation. Les décisions par rapport à l'utilisabilité entraînent des changements de type organisationnel. Ces changements passent à travers la conscience des gens.
Quelles sont les caractéristiques de la participation des développeurs dans cette évaluation ? Une première approche afin de répondre à la question : sommes-nous vraiment centrés sur le client ?, les développeurs peuvent utiliser un questionnaire standard qui est disponible sous demande. Les résultats des évaluations d'utilisabilité sont adressés aux gestionnaires, aux concepteurs et aux développeurs, qui sont ceux qui vont décider et implémenter les changements ou les corrections à faire sur le produit évalué, s'il y a lieu. Toutes les personnes concernées doivent savoir que le propos de l'évaluation n'est pas d'évaluer leurs compétences ou leurs connaissances. La cible de l'évaluation est le logiciel et, notamment, ses caractéristiques en ce qui concerne sa facilité d'utilisation de la part des utilisateurs réels. Également, l'objectif de l'évaluation ne porte ni sur la conception du logiciel ni sur les ressources techniques utilisées, même si à partir des suggestions sorties de l'évaluation des changements dans ce niveau pouvaient être convenables.
Conclusion Les résultats obtenus à partir de l'évaluation d'utilisabilité du logiciel Diamant vont permettre de décider l'implémentation des changements nécessaires sur le logiciel afin d'améliorer l'interaction de l'utilisateur finale du logiciel. La prémisse est que le dernier mot en ce qui concerne le succès d'un produit logiciel appartient à l'utilisateur final et qu'un haut degré de satisfaction subjective de l'utilisateur est la clé de l'évolution de n'importe quel projet logiciel.

Tableau A.6 – Modèle de document d'intention destiné aux évaluateurs

Document d'intention destiné aux évaluateurs
Introduction Le groupe eXit de la Faculté de Génie de l'Université de Sherbrooke travaille depuis quelques années dans un logiciel de construction d'horaires qui fonctionne dans plusieurs facultés de cette Université. La planification et l'implémentation d'une évaluation d'utilisabilité est l'une des étapes considérées d'importance stratégique pour la continuation et l'amélioration de la production de nouvelles versions du logiciel. L'un des membres de l'équipe a présenté un projet de recherche afin de déterminer les concepts de base, d'explorer les principales techniques d'évaluation et de générer un premier ensemble d'outils afin d'évaluer l'utilisabilité des produits logiciels qui seront appliqués dans le sein du groupe de développement. La méthodologie suivie pour atteindre cet objectif doit respecter certaines contraintes de coût et de temps.
Qu'est-ce qu'une évaluation d'utilisabilité ? Une évaluation d'utilisabilité est un test visant à établir si le système mis en œuvre remplit ses fonctions premières telles que déterminées par ses utilisateurs. Une évaluation d'utilisabilité essaie de mesurer les facteurs de qualité d'une interface utilisateur-ordinateur (interaction homme-machine) d'un logiciel. Le concept d'utilisabilité peut être vu depuis différentes perspectives. D'une part, représente la qualité d'un matériel ou d'un logiciel qui est facile et agréable à utiliser et à comprendre, même par quelqu'un qui a peu de connaissances en informatique et/ou en ordinateurs. D'autre part, dans le cas des logiciels utilisés pour travailler, l'utilisabilité signifie la transparence du logiciel en tant qu'outil, qui permet à la personne d'avancer dans son travail et d'augmenter sa productivité, sans introduire plus des contraintes ou des obstacles additionnels que les propres à sa tâche. Finalement, bien que souvent soit utilisé avec le sens de "convivialité", le terme utilisabilité possède un sens plus large qui correspond "à la capacité d'un système à permettre à ses utilisateurs de faire efficacement ce pour quoi ils l'utilisent". Afin que le travail soit fait, le système utilisable doit non seulement être facile à utiliser, mais aussi fiable et efficace. L'utilisabilité est l'un des domaines de l'interaction personne-ordinateur. Elle diminue l'écart entre les gens et les machines. L'interface d'utilisateur ou interface humain-ordinateur se réfère à la partie du matériel et/ou du logiciel qui permet à une personne de communiquer avec un ordinateur. Cela inclut les dispositifs d'entrée (la façon de parler de la personne avec la machine) et de sortie (la façon de parler de la machine avec la personne). Chacun de ces composants d'interface possèdent dispositifs correspondants aux chaînes visuelles, audio et tactiles du cerveau humain. L'utilisabilité étudie ces éléments de l'expérience de l'utilisateur. Par conséquent, l'un des éléments fondamentaux d'une évaluation d'utilisabilité contemple la participation active de l'utilisateur réel.

Tableau A.7 – Modèle de document d'intention destiné aux évaluateurs (cont.)

Quelles sont les responsabilités des évaluateurs ?
La responsabilité de l'évaluateur consiste essentiellement à [5] :

1. Protéger l'utilisateur
2. Préciser à l'utilisateur ce qu'il doit faire (tâches)
3. Encourager l'utilisateur à penser en voix haut
4. Écouter ce que l'utilisateur a à dire ou veut dire

Les consignes sont :

1. Suivre le plan établi (prendre de notes, faire les questions préétablies et noter les réponses) Ne pas intervenir dans l'interaction utilisateur-machine. Ne pas donner des conseils ou de suggestions de comment procéder .
2. Essayer de vérifier en avance le fonctionnement correct du logiciel et l'accès à toutes les données nécessaires.
3. Protéger les participants. C'est la responsabilité de l'évaluateur de prévenir n'importe quel dommage à l'amour-propre et à la dignité de l'utilisateur. Être gentil. S'il se coince, être sûr qu'il n'est pas trop frustré par l'expérience. Remercier sa participation dans l'expérience. Faire savoir que sa participation a été utile et précieuse.
4. Être emphatique : être gentil, patient et rassurant. Préciser que vous savez que l'utilisateur n'est pas stupide.
5. Essayer de capter la pensée de l'utilisateur, les incertitudes, les doutes. La tâche principale de l'évaluateur est d'observer le processus de la pensée de l'utilisateur. Il faut l'encourager à penser à voix haute ou lui demander explicitement ce qu'il pense quand il reste plusieurs seconds en regardant l'écran sans rien faire. (Qu'en pensez-vous ?, qu'en observez-vous ?, etc.) Pendant l'évaluation, il faut mettre plus d'attention sur les explications ou les questions des utilisateurs que sur les opinions personnelles.

Une prémisse très importante à tenir compte est : Personne ne veut pas sembler stupide. Le protocole consiste à :

1. Papier et crayon
2. Enregistrement d'audio
3. Enregistrement de l'activité de l'utilisateur (vidéo, automatiquement par ordinateur, si possible)

Conclusion
Les résultats obtenus à partir de l'évaluation d'utilisabilité du logiciel Diamant vont permettre de décider l'implémentation des changements nécessaires sur le logiciel afin d'améliorer l'interaction de l'utilisateur finale du logiciel. La prémisse est que le dernier mot en ce qui concerne le succès d'un produit logiciel appartient à l'utilisateur final et qu'un haut degré de satisfaction subjective de l'utilisateur est la clé de l'évolution de n'importe quel projet logiciel.

ANNEXE B
MODÈLES DE RAPPORTS

B.1 Introduction

Cette annexe contient les rapports d'évaluation présentés aux développeurs et aux gestionnaires. Outre d'exposer les résultats finaux de l'évaluation faite, ils servent des modèles à propos du contenu et d,exemple de format pour futures évaluations.

Les rapports des sections B.2 et B.3 sont accompagnés par des fichiers xls avec une description détaillée des défauts trouvés.

La section finale B.5 présent le sommaire exécutif destiné aux gestionnaires, où les résultats finaux de l'évaluation sont exposés, les solutions et suggestions à court et long terme sont proposées.

B.2 Rapport d'utilisabilité de Diamant

Ce rapport a pour objectif la présentation des résultats de l'évaluation préliminaire du logiciel Diamant, version 1.2.6.8. Pour commencer, nous allons évaluer la documentation associée au logiciel afin d'avoir une première approche en ce qui concerne les fonctionnalités et l'opération de base du logiciel. Après, nous allons évaluer l'application en tant que révision experte, c'est-à-dire du point de vue d'un expert en utilisabilité qui révise les éléments d'interface utilisateur par rapport à la conformité avec les normes en vigueur. Une prémisse adoptée dans ce travail est que le développeur désigne l'interface d'interaction du logiciel selon son schéma mental et sa perception de ce qui est clair ou non par rapport aux tâches à effectuer. Habituellement ce modèle mental ne coïncide pas avec le modèle mental de l'utilisateur réel du logiciel. Alors, le développeur n'est pas la meilleure personne pour évaluer sa propre interface du point de vue de l'utilisabilité. Il comprend sa propre interface, car il l'a conçue et il trouve comment procéder rapidement, car l'interaction suit son raisonnement, son intuition, sa logique et sa connaissance. Cependant, ces éléments sont rarement similaires à ceux des utilisateurs réels.

B.2.1 Évaluation de la documentation

Objectif

Le but de cette section est de présenter les défauts du point de vue de l'utilisabilité du logiciel, en ce qui concerne la documentation fournie avec le produit. Cette documentation

consiste à un manuel d'utilisateur qui comprend 72 pages divisées en quatre sections et deux annexes. Il faut considérer que nous allons lire tout le manuel d'utilisateur avant de commencer les preuves sur l'application ce qui ne s'applique pas aux utilisateurs réels. Ils essaient habituellement d'utiliser un logiciel intuitivement et seulement consultent le manuel en cas de problèmes ou quand ils ne peuvent pas comprendre pourquoi ce qu'ils font ne marche pas.

Liste des défauts

La liste des défauts généraux et détaillés se trouve dans le fichier adjoint ListeDéfauts_V01_Diamant.xls. Les sous-sections suivantes décrivent d'une façon globale les défauts relevés à travers les différentes parties de la documentation.

Documentation-Chapitre 2 : Installation et mise à jour

La procédure d'installation est inadéquate d'après le profil d'utilisateur de l'application. Elle présente beaucoup de concepts de bas niveau, faciles à comprend par les développeurs, mais peu utiles à l'utilisateur réel. La description des chemins (paths), versions, noms de fichiers en outre d'être non familiers et dépourvus de signification à l'égard de l'utilisateur, risque de devenir obsolète rapidement et, par conséquent, inexacte. Il faudrait développer une installation automatique, facilement reproductible et vérifiable où les détails de bas niveau soient cachés à l'utilisateur.
Note : L'adresse "http ://gap2.gel.usherb.ca/diamantbug" qui apparaît dans la documentation est inexistante.

Documentation-Chapitre 3 : Modification de la grille horaire

Il faudrait être sûr que l'utilisateur comprend bien les éléments avec lesquels il doit travailler. Par exemple : quelle est la différence entre une grille horaire et un horaire ?, que signifie la priorité d'une période ? La définition précise avant de commencer la description des fonctionnalités donnera un sens aux différents éléments de l'interface et à l'ordre des étapes de travail qui restera dans la mémoire à long terme de l'utilisateur. Il faudrait d'abord définir toute la terminologie employée (par exemple : événement, activité, nature, groupe, bloc, etc.), expliquer les éléments qui apparaissent sur l'écran et, en particulière, comment interpréter l'écran complet de la grille (abréviations, couleurs, numéros, etc. gris clair indique désactivation, inhibition, etc.) qui est l'élément principal du logiciel.

Documentation-Chapitre 4 : Construction d'un horaire

Un problème détecté est le manque de consistance à travers de l'interface. Cela s'étend sur les boutons (légendes et positions), titres de fenêtres, interaction dans les fenêtres de dialogue d'ouverture, de sauvegarde de fichiers et l'utilisation des allusions (hints). L'utilisateur ne lit pas tout le temps les légendes (captions) de boutons ou de fenêtres, il automatise la procédure et les mouvements, il clique par reflet sur les emplacements. Le manque de consistance augmente la probabilité d'erreur. Ce type d'erreur est gênant, car il empêche de suivre

un rythme de travail continuel, interrompre la mémoire à court terme, produit des délais, diminue la productivité (temps et erreurs). Ces erreurs sont évitables. En ce qui concerne les titres des fenêtres, il est conseillé d'utiliser une expression qui représente l'opération que l'utilisateur peut réaliser à partir de cette fenêtre. Dans le cas du logiciel sous évaluation, parfois il s'agit d'un titre que ne représente pas l'opération (Nom : Grille horaire), parfois il est une instruction à l'utilisateur (Verbe en mode impératif : Sélectionnez une option), parfois il est l'indication d'une opération (Verbe en infinitif : Définir fichiers pour l'importation automatique). Parfois il y a une référence type énigme (Affectation XXX). Par rapport aux fenêtres de dialogue pour ouvrir et enregistrer un fichier, les titres des fenêtres ne sont pas consistants. Parfois il est utilisé un nom (Grille horaire), parfois un verbe en infinitif (Enregistrer). De manière générale pour fermer n'importe quelle fenêtre il devrait se faire en utilisant un seul mot et une position fixe sur la fenêtre. Dans le cas du logiciel sous évaluation, parfois il est utilisé le mot fermer, parfois une fenêtre est fermée en appuyant sur d'autres boutons (le mot annuler sert à fermer une fenêtre aussi). Les boutons Modifier, Appliquer parfois ne sont pas toujours significatifs. En ce qui concerne les rapports, ils sont peu clairs quant à l'information fournie et surtout, à la manière de l'utiliser pour prendre de décisions. Quand le nombre de files est très grand (grand nombre de conflits, par exemple) il est très difficile d'extraire de l'information significative rapidement. Un autre aspect est le manque de raccourcis (shortcuts) pour les boutons et les items de menu. Cela constitue un standard d'interaction qui est utilisé par les utilisateurs dans la mesure qu'ils deviennent experts dans le logiciel et qui aide à augmenter leur rythme de travail.

Documentation-Annexes

Une question se pose face au contenu des fichiers : Pour quoi l'utilisateur a-t-il besoin de connaître la structure de fichiers ? Il ne construit pas ces fichiers, il ne doit pas intervenir en leur construction, il ne doit pas les comprendre. Dans le cas où la construction de ces fichiers est partie de la tâche de l'utilisateur du logiciel Diamant, il faudrait concevoir une application spécifique avec une interface convenable pour générer ces fichiers correctement et permettre d'entrer de valeurs valides à l'utilisateur. Les annexes A et B contiennent des données codifiées de trop bas niveau, ils sont adressés au développeur, mais non à l'utilisateur normal. Ce n'est pas adéquat d'inclure cette information dans le manuel d'utilisateur.

Conclusion de l'évaluation de la documentation

Les problèmes principaux détectés dans la documentation concernent au manque de consistance, le manque de respect aux standards d'utilisabilité, l'emploi d'une terminologie de bas niveau et d'une terminologie spécifique des tâches non définie.

La documentation doit être conçue comme un matériel instructif, cela veut dire, d'enseignement. Par conséquent, elle doit suivre l'ordre naturel des concepts, de la terminologie et des tâches selon le profil d'utilisateur. Cela donne une signification aux différents éléments et étapes, et par conséquent, rend l'information facilement compréhensible et mémorisable.

Un type de documentation détaille toutes les fonctionnalités plus ou moins en ordre des tâches de l'utilisateur. Un autre type de documentation (tutoriel) décrit pas à pas com-

ment faire dès le début à la fin, un ensemble de tâches basiques. Dans le deuxième cas, il est possible de décrire, par exemple, comment construire un nouvel horaire à partir de données réelles dès "Nouvelle Grille" jusqu'à ce que l'horaire soit fini, c'est-à-dire, sans la présence de conflits et sauvegardé.

Si les règles de consistance sont connues et respectées dès le départ, il s'agit de petits détails faciles habituellement à remplir, spécialement en ce qui concerne la consistance à niveau design d'interface. Mais, à la fin d'un projet, ce sujet est un aspect gênant tant pour le développeur que pour l'utilisateur.

B.2.2 Évaluation de l'interface du logiciel

Objectif

Le but de cette section est de lister les défauts pertinents sur le logiciel concret. Pour cela, nous allons essayer de construire un horaire et d'examiner les différentes fonctionnalités de l'application.

Approche initiale

Nous avons essayé de construire un horaire nouveau à partir des idées apprises pendant la lecture du manuel d'utilisateur. L'information sémantique présente dans la mémoire à long terme était :

1. Il fallait disposer de certains fichiers xml de cours, étudiants, enseignants, locaux.
2. Il fallait les "définir" (choisir, déterminer) et "importer" dans l'application
3. Il fallait faire une première assignation d'horaire en utilisant la fonction "Affectation Initiale"

Après avoir réalisé ces étapes, nous étions en présence d'une grille avec des couleurs sans définitions par rapport à la signification et d'une série de valeurs numériques parmi lesquelles le plus significatif était l'un qui indiquait qu'il y avait plus de 400 conflits, plus de 380 conflits d'étudiants. Il n'y avait pas une aide pour mieux comprendre ce qui était possible de faire et pour visualiser les conflits afin de choisir une stratégie de résolution. Nous avons essayé de trouver une piste à partir des rapports. Encore une fois, nous avons trouvé de l'information codifiée qui ne nous permettait pas de comprendre vraiment les conflits et d'envisager un possible chemin vers la solution. Nous avons décidé, alors, de consulter le manuel d'utilisateur. La phase 2 (page 12) de "modification et épurations de données" ne nous a pas aidés, car il est peu utile s'il n'y a pas une idée claire sur la source du problème. La même situation avec les phases 3 et 4. Nous avons alors essayé de parcourir tout le manuel afin de trouver quelque chose qui nous permettait d'augmenter la compréhension de ce qui nous présentait l'interface du logiciel et de cette façon avancer le travail. Le manuel n'a pas une section type tutoriel qui exemplifie une séquence complète de comment accomplir une tâche de construction d'horaire à partir de données réelles. Il n'existe pas un tutoriel imprimé ou en ligne. Probablement une meilleure compréhension initiale s'obtiendrait avec

un affichage différent de la grille tel que le mode détaillé (Préférences - Détaillé + split), mais cette fonctionnalité n'est pas mentionnée dans le manuel. Nous avons trouvé la section de "recherche de potentiels conflits" dans la page 16. Il faudrait avertir de cette possibilité avant. Le rapport de conflits détaille les problèmes des étudiants par heure, c'est-à-dire que le même étudiant apparaît plusieurs fois, même si le conflit ne pourra pas se régler par heure. Le niveau de détail est ainsi peu pratique afin de prendre des décisions. À ce moment-là, une question aux développeurs s'est imposée : Quelle est la procédure à suivre pour construire un horaire avec des données réelles qui comportent centaines de conflits ? Nous avons décidé d'exclure plus de la moitié des activités et de cette façon nous avons éliminé "artificiellement" le nombre de conflits à moins de 10. À partir de cela, nous avons réglé les conflits des enseignants et des étudiants par le biais de l'utilisation de la fenêtre d'Affectation manuelle et la visualisation de conflits potentiels en couleurs. Cependant, nous sommes arrivés à un moment où les conflits de locaux ne pouvaient pas être réglés. En termes généraux, la visualisation graphique de conflits potentiels est un outil très utile afin de replacer des activités pourvu que le nombre de conflits soit petit. Jusqu'à présent, nous avons essayé le logiciel avec des données apparemment correctes, avec des fichiers compatibles et correctement construits (les fichiers exemples fournis). Il faudrait analyser la réponse du logiciel avec des fichiers qui contiennent des erreurs de données ou des incompatibilités variées. Également, jusqu'à présent, nous avons essayé le logiciel avec des données correspondantes à la faculté de génie. Il faudrait essayer les autres facultés.

Liste des défauts

La liste des défauts généraux et détaillés se trouve dans le fichier adjoint "ListeDefauts_V01_Diamant.xls".

Conclusion de l'évaluation de l'interface du logiciel

Le design d'un aide en ligne serait un avantage qui peut ajouter une grande valeur au logiciel. Il devrait être désigné de façon de remplir deux besoins : un accès rapide aux détails syntactiques et aux éléments partiellement oubliés à cause du manque de fréquentation du logiciel et une explication plus détaillée, mais conçue par fragments et liens qui permettent le dosage de l'information nécessaire pour le nouvel utilisateur.

La fonctionnalité "défaire" (undo) devrait s'implémenter au moins dans le cas de modifications des items de menu "Affectation". De cette manière, il est possible d'essayer des changements, d'observer les résultats par rapport aux conflits et de décider en conséquence.

Par rapport aux couleurs, le logiciel utilise au moins 10 couleurs dans les formes et les caractères utilisés, ce qui peut être trop du point de vue de la signification comme élément de codification. D'ailleurs, il n'est pas respecté la codification naturelle (rouge indique danger, interdiction - jaune indique avertissement, faire attention, être alerte - vert indique condition normale, permission, etc.). En tout cas, la signification devrait être précisément définie et indiquée explicitement de manière simple, non distractive, sans consommation appréciable d'espace sur l'écran.

Il faudrait penser à d'autres façons de représenter graphiquement l'information afin de permettre d'identifier plus rapidement les sources de conflits et la prise de décisions dans le but de les éliminer. Également, des changements par manipulation directe sur des graphiques et l'implémentation des fonctionnalités "défaire" (undo) et "refaire" (redo) fourniraient des outils précieux à l'heure de faire face à un grand nombre de conflits. La première, car il résulte plus immédiat de suivre un problème d'emplacement d'événements de manière graphique et par manipulation directe des objets sur l'écran. La deuxième, car elle permet à l'utilisateur d'essayer et de revenir à l'état antérieur rapidement au cas où le changement ne serait pas convenable. La correction de ce dernier défaut implique des changements majeurs de l'application et par conséquent, demande beaucoup plus de temps de réalisation.

B.2.3 Conclusion générale de l'évaluation de Diamant

La plupart des défauts détaillés dans le document xls adjoint sont assez simples de corriger. Il faudrait régler les problèmes de consistance par le biais de l'adoption d'une stratégie de codification et de présentation, la création d'un document de déclaration et définition et après, l'implémentation sur le logiciel.

Une application d'installation automatique aurait un impact positif sur les nouveaux utilisateurs et chaque fois qu'il faut mettre à jour ou modifier l'installation de l'application.

Une aide en ligne, plus qu'un atout, est un impératif étant donné le profil de fréquentation de l'utilisateur. Ce type d'aide se divise, au moins, en aide proprement dit, aide sensible au contexte et allusions (hints). Étant donné le profil d'utilisateur, un rappel des éléments codifiés (abréviations, couleurs, etc.) sous la forme d'aide sensible au contexte et d'allusions ou de petites étiquettes (labels) l'aideraient à se souvenir des aspects partiellement oubliés. Par contre, l'aide proprement dit, organisé et complet s'adresse à un utilisateur nouveau.

Le style d'interaction prédominant dans l'interface est la sélection par menu et l'utilisation de contrôles (spin edits, list boxes, etc.) qui préviennent l'erreur d'utilisateur. Cependant, dans la construction de l'horaire et surtout dans la tâche d'élimination de conflits, il faudrait penser en l'utilisation du style de manipulation directe , afin d'essayer le replacement des activités plus agile.

La documentation imprimée (mieux en ligne si possible) devrait ajouter un tutoriel pour les utilisateurs nouveaux. Cela permet de rapidement commencer à travailler avec le logiciel et obtenir des résultats visibles, ce qui a un impact très positif sur la satisfaction subjective de l'utilisateur.

Il faudrait éliminer toute nécessité de connaissance de bas niveau. Les utilisateurs ne devraient pas avoir besoin de connaître la structure de bas niveau des fichiers nécessaires à la construction des horaires. Cela implique la création d'une application avec une interface adéquate au profil d'utilisateur ou bien, un module de vérification et de validation des données qui rapportent les erreurs trouvées et/ou essaie de les corriger et/ou guide l'utilisateur à propos de comment régler la situation.

B.3 Rapport d'utilisabilité de Diamant Web

Ce rapport a pour objectif la présentation des résultats de l'évaluation préliminaire du logiciel DiamantWeb, version courante sur Internet depuis le 25 septembre 2006. Ce travail est une continuation du rapport d'évaluation d'utilisabilité du logiciel Diamant, déjà présenté. Un détail de la définition de concepts de base et de la terminologie a été présenté dans le rapport mentionné. Par conséquent, il est conseillé de réviser les sections 2 (Profil de l'utilisateur et de tâches) et 3 (Conception de l'interface utilisateur) avant de procéder a la lecture du présent document. Comme déjà a été dit dans le rapport cité ci-dessus, une prémisse adoptée dans ce travail est que le développeur désigne l'interface d'interaction du logiciel selon son schéma mental et sa perception de ce qui est clair ou non par rapport aux tâches à effectuer. Habituellement ce modèle mental ne coïncide pas avec le modèle mental de l'utilisateur réel du logiciel. Alors, le développeur n'est pas la meilleure personne pour évaluer sa propre interface du point de vue de l'utilisabilité. Il comprend sa propre interface, car il l'a conçue et il trouve comment procéder rapidement, car l'interaction suit son raisonnement, son intuition, sa logique et sa connaissance. Cependant, ces éléments sont rarement similaires à ceux des utilisateurs réels.

B.3.1 Objectif

Le but de cette section est de présenter les défauts du point de vue de l'utilisabilité du logiciel, en ce qui concerne l'application et la documentation fournie. Cette documentation consiste à deux écrits de 40 pages, 3 chapitres et 20 pages, 3 chapitres respectivement, établis comme documents de guide pour l'utilisateur du logiciel DiamantWeb. Étant donné que la fonctionnalité essentielle du logiciel est simple si l'application Diamant est connue, nous avons essayé de faire un ensemble minimum de tâches basiques (voir dans l'écran une grille, par exemple) du logiciel à partir d'information verbalement donnée, sans lire d'abord la documentation. C'est-à-dire, nous avons imité le comportement naturel de l'utilisateur qui déjà connaît Diamant, qui consiste à utiliser comme guide l'intuition. Mais, nous n'avons pas réussi. C'est pour quoi, nous avons recommencé par la lecture de la documentation. Il faut considérer que nous allons lire tout le manuel d'utilisateur avant de recommencer les preuves sur l'application ce qui ne s'applique pas aux utilisateurs réels. Ils essaient habituellement d'utiliser un logiciel intuitivement et seulement consultent le manuel en cas de problèmes ou quand ils ne peuvent pas comprendre pourquoi ce qu'ils font ne marche pas. Cela s'applique surtout aux utilisateurs connaisseurs ou experts.

B.3.2 Liste des défauts généraux

Page d'accueil

La page d'accueil ne suit pas les normes de design d'interface d'utilisateur. La page d'accueil est une page de présentation où l'information s'oriente sur le propos de l'application et les fonctionnalités d'abord, et après sur des instructions. Il n'est pas conseillé de mettre au premier plan des instructions à propos de choses que l'utilisateur ne connaît pas encore.

D'ailleurs, les instructions aux utilisateurs placés dans des grilles, des couleurs mal employées et des références ou liens aux manuels qui ne marchent pas ne sont pas convenables et moins, dans la page d'accueil.

Gestion de grilles

Nous avons essayé de construire une grille à partir de zéro. La création de chaque jour, chaque séquence, chaque période un par un, signifie passer au moins pour trois fenêtres et cinq cliques (l'une desquelles est un message à toute fenêtre qui indique que l'opération est réussite) par chaque ajout d'un élément. Cette interaction ne respecte pas la logique de l'utilisateur qui cherche de définir les éléments répétitifs d'une seule fois. Le concept de niveau peut être convenable du point de vue de la logique de programmation, mais la navigation n'a pas été résolue efficacement, car, du point de vue de l'interaction, il est difficile de reconnaître rapidement où on est placé dans un moment particulier. Il faudrait clarifier dans quel niveau on se trouve d'une manière graphique puisque les titres sont très longs et il faut lire pour se situer. Le design des fenêtres ne respecte pas les normes de distribution des composants sur l'écran, il est nécessaire d'utiliser souvent les bars de défilement horizontal pour voir la grille au complet en tant que la plus grande partie de la fenêtre est vide (scroll bars). Les couleurs utilisées sont inappropriées tant du point de vue de la codification de couleurs que de l'ergonomie. Dans le premier cas, ce sont beaucoup de couleurs avec des significations similaires à travers de toute l'interface. S'il est nécessaire de consulter tout le temps la signification de la codification des éléments, cela implique qu'elle ne sert pas à simplifier la tâche pour laquelle elle a été implémentée. Dans le deuxième, les couleurs verts, bleus et bordeaux foncés l'un sur l'autre sont illisibles. Il existe des normes pour utiliser des pairs de couleurs permises (fond et texte, premier plan sur fond, etc.) étudiés et définis en fonction de la lisibilité et de la minimisation de la fatigue visuelle. Ces normes ne sont pas respectées. Les légendes de links, pseudo-boutons contiennent des verbes qui ne sont pas familiers à la tâche de l'utilisateur. L'utilisateur veut "créer-voir-modifier-sauvegarder" une grille. Les actions telles que "télécharger-exporter-importer-soumettre" peuvent être confuses. Le manuel n'aide pas à bien comprendre la signification de ces opérations. D'ailleurs, il faut tenir compte la compatibilité et la consistance de la terminologie avec le logiciel Diamant. Il faudrait tenir compte que la terminologie des ordinateurs est utilisée par des gens de manière non technique, intuitive et habituelle sans savoir exactement ce qu'elle signifie. C'est pourquoi le manuel doit être une documentation instructive, c'est-à-dire, le contenu doit enseigner, guider, expliquer. Les explications des commandes et des fonctionnalités doivent être instructives. Par exemple,

Fonction	Importer
Explication	Cette fonction vous permet d'importer une grille horaire d'un fichier

Cette explication ne dit rien, car le même mot qu'on essaie d'expliquer est utilisé dans la définition. Si l'opération implique 3 composants : entrée - action - sortie, elle doit être définie par tous ces composants clairement définis et précisés, en utilisant des mots familiers à la tâche de l'utilisateur. La séquence de tâches naturelle et les noms des actions ne respectent

pas l'ordre naturel de la tâche de l'utilisateur. Par exemple, pour voir (afficher) une grille déjà existante, il faut cliquer sur le lien "modifier" et après, dans une fenêtre étrange à la tâche, cliquer sur "afficher". Un processus naturel serait de cliquer sur le nom de la grille, mais cette action renvoie à une fenêtre "Modifier une grille horaire". Convenons que, aller à quelque chose qu'indique "Modifier" quand on essaie seulement de "Voir" peut être perçu comme dangereux. L'opération d' "Effacer" est clairement présentée par l'interface comme dangereuse dans le sens qu'il n'est pas possible de défaire. L'inconsistance est que pour confirmer ou non l'opération, l'action par défaut (lien bleu traditionnel) est "Oui (Accepter)" en tant qu'il devrait être l'action moins préjudicielle. L'action de cliquer sur "Télécharger" dans la fenêtre de "Gestion de Grilles" montre un fichier xml. Cette structure est d'utilité seulement pour les développeurs, non pour les utilisateurs selon le profil standard défini. Il faudrait se demander quelle est l'utilité pour l'utilisateur. Le manuel ne fait pas référence à cet aspect. Les contrôles de type "spin edit" sont utiles quand le nombre d'items est près de dix, plus de cette quantité implique une interaction trop lente et très susceptible d'erreur.

Disponibilité

La spécification des unités avec les numéros aide à la compréhension. Par exemple, "30 min" permet plus rapidement de comprendre qu'il s'agit d'une période que si seulement le numéro 30 est montré. La procédure d'exporter, la création des fichiers ".dia" s'avère assez complexe du point de vue d'un utilisateur standard (section 2.6 du manuel). Le contenu syntactique de cette tâche est inapproprié. Il est fort probable que personne ne se rappelle cette procédure. Par conséquent, l'erreur est probable et prévisible. Dans le manuel apparaissent les concepts de disponibilité générale de faculté et de départements, mais la forme de les afficher ne les fait pas évidents. L'utilisation de mots tels que "Propagation" et "Exception" paraissent plus appropriés aux développeurs qu'aux utilisateurs. Sans une référence concrète dans le manuel, ils ne seront pas déductibles. Il faudrait revoir la manière de présenter cette fonctionnalité. L'interaction dans les fenêtres de disponibilité de jours de facultés et de départements est incorrectement définie. Le comportement naturel serait de choisir (cliquer pour sélectionner) un jour, ou période et après, indiquer (cliquer sur la fonction) le type de priorité qui est demandée pour ce jour ou cette période. La disponibilité de locaux n'a pas été évaluée dû à une erreur "Invalid Path http Error Status 400". Également, l'erreur "Servlet exception" a apparu plusieurs fois dans les sessions d'évaluation.

Annexe

Les guides rapides sont consultés par l'utilisateur quand il a oublié quelque détail de comment faire une certaine tâche qu'il a réalisé auparavant et qu'il connaît sémantiquement. S'il veut "modifier" ou "ajouter" quelque chose et il ne parvient pas à le faire, un guide que dit "cliquer sur modifier" ou cliquer sur "ajouter" sûrement ne lui sera pas utile. L'ordre de tâches est présenté de manière incompatible avec l'ordre de taches de l'utilisateur. Par exemple, afficher une grille apparaît dans le 7ème lieu. Le concept de "Créer une grille" qui appartient au domaine de la terminologie de l'utilisateur, a été représenté par "Ajouter une grille" ce qui peut ne pas être évident.

B.3.3 Conclusion générale de l'évaluation de Diamant Web

Comme conclusion de l'évaluation d'utilisabilité réalisée la recommandation finale comprend deux parties, la première implique de changements majeurs dans le logiciel, la deuxième comporte des changements d'impact mineur. En ce qui concerne la première partie, nous suggérons une ré-conception de l'interface d'utilisateur afin de permettre de construire une nouvelle grille d'une façon logique en minimisant les interactions répétitives. Par rapport à la deuxième partie, nous suggérons

1. Utilisation de la terminologie en accord avec la tâche de l'utilisateur
2. Définition précise de la terminologie, surtout s'il y a des éléments étranges à l'utilisateur
3. Révision de la navigation : l'utilisateur doit toujours savoir où il est de manière instantanée sans devoir lire de longs titres (concept de niveaux)
4. Instructions à l'utilisateur : les links à l'aide ou au manuel doivent toujours fonctionner
5. Générale : révision complète de l'utilisation de la couleur, de la distribution des éléments dans la fenêtre et la correction des erreurs d'orthographie, de majuscules/minuscules et de mélange de langues
6. Tous les éléments tels que changements d'extension de fichiers, présentation de structures xml, etc. (qui représentent des concepts syntactiques et de bas niveau) devraient disparaître d'une nouvelle version

B.4 Rapport des observations directes de l'utilisateur

À manière d'exemple, deux rapports d'observation directe de l'utilisateur sont reproduits tout de suite.

Introduction

Ce rapport a pour objectif la description des observations faites pendant le travail des utilisateurs des logiciels Diamant et DiamantWeb dans la Faculté d'Éducation Physique (tableau B.1) et dans la Faculté de Sciences (tableau B.2).

B.4.1 Rapport 1 : Faculté d'Éducation Physique

Description des activités

D'abord, l'utilisateur a montré toute la documentation pertinente préparée par lui-même afin de mettre à jour les données pour la session à travailler (dans ce cas la session d'hiver 2007). Ces documents comprennent des fichiers .doc, .xls, e-mails, calendrier, afin de définir les particularités telles que les caractéristiques de chaque cours, la disponibilité des enseignantes, les groupes par cours et les communications aux professeurs.

L'utilisateur utilise la fonctionnalité de disponibilité des enseignantes comme partie de son travail.

Tableau B.1 – Rapport d'observation - Faculté d'Éducation Physique

Item	**Description**
Observation N°	1
Date Heure	1er Novembre 2006 - 8h30 à 10h00
Lieu	A8-204
Utilisateur	...
Observateur	Miriam Berro
Duration	1h30
Tâche	Préparation des données pour travailler l'horaire avec Diamant
Sous-tâche1	Vérifier la mise à jour des données des professeurs dans Diamant-Web
Sous-tâche2	Importer-Exporter le fichier project_horaire.xml dans Diamant Web pour obtenir le fichier .dia utilisé par Diamant. Ce fichier .xml contient les pre-choix des étudiants.

L'utilisateur reçoit un fichier avec le pre-choix des étudiants de la part de la secrétaire qui rentre les données au SIIG. Ce fichier s'appelle projet_horaire.xml. Ce fichier montre 489 étudiants, donnée qui sera confrontée avec les données produites par Diamant, après la création du fichier .dia.

L'utilisateur a déjà fait ses grilles horaires il y a longtemps et pour l'instant il n'a pas besoin de créer de nouvelles grilles.

L'utilisateur dispose d'une guide rapide d'une seule feuille en papier faite par les développeurs et ses propres notes sur les étapes à suivre pour accomplir la tâche ci-dessus décrite, intercalées dans le manuel d'utilisateur.

L'utilisateur a mentionné qu'il serait très utile pour lui de disposer de ce type de guide ponctuel et rapide dans DiamantWeb, pour faciliter et assurer l'accès.

Un autre aspect mentionné est que le manuel explique une fonctionnalité isolée, mais pas comment enchaîner plusieurs fonctionnalités afin de faire une sous-tâche concrète. D'ailleurs, les explications pour le cas de construire un horaire de sessions ou un horaire d'examen sont mélangées et parfois confuses. Malgré les similitudes, l'explication devrait se répéter pour chaque cas.

Travail sur DiamantWeb

L'accès aux documents d'aide n'a pas marché.

La vérification de la mise à jour de la base de données de professeurs s'est effectuée dans DiamantWeb, Disponibilité, Enseignantes. L'utilisateur est allé voir quelques noms de nouveaux professeurs qu'il savait par coeur. Cela a donné comme résultat que, les professeurs

à ajouter n'étaient pas visibles, alors la base de données n'était pas actualisée et le travail ne pouvait pas continuer, au moins, il ne valait pas la peine de le continuer.

L'utilisateur a appelé par téléphone au chef du groupe et au laboratoire de développeurs. Comme il n'y avait personne de disponible, il a envoyé un courriel.

Il a mentionné qu'étant donné qu'il commence de travail plusieurs jours avant la date limite, il peut le retarder un jour ou deux sans problèmes majeurs. Mais, bien sûr, si le cas était autre et il devait terminer son travail pour le jour suivant, il serait vraiment très contrarié.

L'utilisateur a commenté que la possibilité d'ajouter en DiamantWeb un professeur lui-même, débloquerait son travail.

L'utilisateur a essayé de toute façon de suivre les étapes jusqu'à disposer du fichier .dia : Disponibilité - Importer projet_horaire.xml - Associer Grille - etc. à partir du manuel avec des notes propres ajoutées.

Les processus dans DiamantWeb prennent beaucoup de temps, mais un bar de progrès est montré, alors l'utilisateur fait une autre tâche en attendant la finalisation des importations, exportations et téléchargements successifs.

Travail sur Diamant

Une fois les étapes apparemment réussies dans DiamantWeb, l'utilisateur a essayé d'ouvrir un horaire dans Diamant. Ce premier essai a donné comme résultat un "hourglass" sur l'écran et après 1 minute l'utilisateur a décidé de terminer Diamant avec CRTL-ALT-DEL.

L'utilisateur a pensé que parfois il a commis une erreur dans les pas antérieurs depuis DiamantWeb. Alors, il a essayé deux fois de plus. Dans la troisième il a obtenu un message d'erreur qu'il ne peut pas interpréter.

L'utilisateur a réfléchi sur les différences entre les instructions de la documentation et ce qu'il regarde dans sa propre machine, tels que "sauvegarder sous" au lieu d' "enregistrer sous".

La session d'observation a fini sans atteindre le but proposé.

B.4.2 Rapport 2 : Faculté de Sciences

Description des activités

D'abord, l'utilisateur vérifie s'il y a une nouvelle version des logiciels, des nouvelles dans Diamant Bug et la documentation. Mais, pour l'instant, les liens ne marchent pas (Manuels, DiamantBug, etc.). Dans le site du group eXit les liens ne marchent pas encore. L'utilisateur n'utilise pas la fonctionnalité de disponibilité des enseignantes comme partie de son travail. L'utilisateur reçoit un fichier projet_horaire_hiver2007.xml. L'utilisateur dispose d'une grille horaire hiver07.xml. 2.1.3

Tableau B.2 – Rapport d'observation - Faculté de Sciences

Item	Description
Observation N°	2
Date Heure	7 Novembre 2006 - 13h30 à 15h00
Lieu	D3-1044
Utilisateur	...
Observateur	Miriam Berro
Duration	1h30
Tâche	Préparation des données pour travailler l'horaire avec Diamant
Sous-tâche1	Importer-Exporter le fichier project_horaire.xml dans Diamant Web pour obtenir le fichier .dia utilisé par Diamant. Ce fichier xml contient les pre-choix des étudiants. L'utilisateur vérifie rapidement ce fichier avec l'application conText.
Sous-tâche2	Ouvrir l'horaire dans Diamant et vérifier les conflits. Obtenir une impression du rapport de conflits.

Travail sur DiamantWeb

Pour commencer l'importation-exportation de project_horaire il dispose des pages 11 et 12 du manuel, section 2.6 Exporter la disponibilité (Création .dia), avec des notes propres où il a signalé qu'il faut importer le projet-horaire.xml.

Comme l'utilisateur n'utilise pas la fonctionnalité de disponibilité, il a observé la fonction importer depuis gestion de grilles. Il a essayé d'importer à partir de la fonction "importer" qui apparaît sur gestion de grilles. L'application a présenté une fenêtre pour définir le fichier, après elle a présenté le bar de progrès, depuis quelques minutes le bar a disparu et la fenêtre reste sans le nom du fichier. Il n'y a pas de message d'opération réussie. L'utilisateur retourne à la fenêtre antérieure et continue.

L'utilisateur voulait avoir une grille avec le nom de la session d'hiver. Il a ajouté une grille, mais après il a voulu la supprimer. Il n'a pas vu la fonctionnalité "effacer". Cette fonctionnalité apparaît dans différentes positions selon la fenêtre soit Gestion de grilles ou Disponibilité.

L'utilisateur prend ces notes et la section 2.6 du manuel, qui est déjà préparée à part, et commence à suivre les pas. L'utilisateur cherche "sauvegarder sous", mais l'option présentée est "enregistrer sous", il croit qu'il s'agit de la même fonction. L'utilisateur ne remarque pas le changement d'extension du fichier. Dans gestion de grilles, essaie d'exporter, mais "télécharger" n'est pas visible. Il essaie encore une fois. L'utilisateur ouvre Diamant et essaie la fonction "Ouvrir horaire", mais les données présentées ne sont pas correctes. Il essaie toute la procédure encore une fois. Il pense avoir oublié quelque chose. D' abord cette fois il peut donner un nouveau nom à une grille déjà existante. Il regarde d'autres pages du manuel. La section 2.5 "importer des données" du SIIG ne l'aide pas. Dans aucun moment

le manuel ne dit comment arriver à la fenêtre "importer une session". L'utilisateur ne peut arriver à cette fenêtre. Il se demande si la version de manuel dont il dispose est la dernière. L'observateur lui offre la dernière version.

Il n'était pas sûr si chaque fois qu'il travaille il doit exporter la grille. Exporter une grille de cours demande beaucoup de temps. L'utilisateur se distrait et perd la séquence des événements.

L'utilisateur a essayé 3 fois la procédure, jusqu'à l'indication de la part de l'observateur d'importer (dans la grille présentée dans la fenêtre) depuis disponibilité. Il a fait de nouvelles notes, sur les pages de la section 2.6 du manuel. Avant chaque essai il a détruit tous les fichiers crées dans l'essai antérieur et vérifie ses répertoires. Avant chaque essai il vérifie la grille. Après le troisième essai l'utilisateur indique que dans le vrai cas, le pas suivante était d'appeler au soutien technique.

Travail sur Diamant

L'utilisateur a ouvert l'horaire .dia depuis Diamant, l'opération a fonctionné. Les données sont correctes. Le nombre de conflits est supérieur à 700. L'utilisateur consulte le rapport de conflit et prépare une impression afin de traiter le problème de conflits à la main avec ses collaborateurs. Il exclut deux activités. En ce qui concerne les conflits, les groupes d'abord sont le principal problème.

Commentaires après cette session

Certaines erreurs qui apparaissent dans une console noire, mais sans la présence de messages ou avertissements depuis Diamant, l'ont fait perdre des heures de travail. Il croyait qu'il enregistrait le travail, mais à cause d'un problème de mémoire, l'application n'enregistrait rien.

Disponibilité de professeurs n'est pas utilisée. Les locaux sont réservés avec Sirus, après le travail avec Diamant terminé. Il ne traite jamais les problèmes de locaux avec Diamant.

L'utilisateur dispose des grilles de cours et d'examens. Quand il a essayé de faire ses grilles d'examen, il a trouvé assez complexe de savoir où il était à chaque instant dans l'interface, il a trouvé que le nombre de pas et de noms était élevé, il a trouvé confuse savoir où cliquer pour chaque action (sur le numéro, sur le nom, etc.), il a mal interprété la fonction "copier", il a manqué de mettre un nom aux jours (lundi, mardi au lieu de 1, 2, etc.), ce qui lui a fait perdre 2 heures de travail dans le passé.

La couleur vert et bleu foncé n'est pas lisible. Les horaires d'examen sont faits pratiquement à la main.

B.4.3 Conclusion des observations

Cette section détaille les éléments d'interface d'interaction additionnels rapportés parmi les observations faites, qui s'ajoutent à la liste de la révision experte déjà réalisée et

présentée dans les rapports d'évaluation de Diamant et DiamantWeb.

- Messages d'erreur sans utilité pour l'utilisateur, dans un ton inadéquat (Diamant-Fatal problem, etc.) et qui utilise des termes étranges au profil de l'utilisateur et au profil de la tâche.
- Messages d'erreur dans une console noire (Diamant), qui n'est pas dans le "focus" du travail de l'utilisateur.
- Aide à l'utilisateur qui ne décrit pas comment réaliser une tâche complète. Il faudrait concevoir une guide pas à pas des tâches les plus fréquentes.
- Fonctionnalité de création d'une grille avec une interaction conçue de façon telle qui implique trop de pas et ne profit pas des caractéristiques répétées qui peuvent s'automatiser. Il faudrait proposer une maquette de nouvelle interaction.

B.5 Sommaire exécutif destiné aux gestionnaires

Ce document a pour objectif la présentation d'un sommaire exécutif sur l'évaluation d'utilisabilité jusqu'à présent des logiciels Diamant et Diamant Web produits par le groupe de recherche εχit sous la direction de M. Ruben Gonzalez Rubio.

L'étude a été commandée à cause des blocages réitérés dans le travail des utilisateurs quand ils veulent traiter les données pour la construction des horaires avec les logiciels cités.

En ce qui concerne les méthodes d'évaluation et d'analyse, nous avons pris comme référence fondamentale les normes de qualité d'utilisabilité établies par les standards ISO9126 et 9241. Les techniques utilisées ont été la révision experte par un spécialiste en utilisabilité et l'observation des utilisateurs pendant son travail avec les logiciels. Il reste l'étape de questionnaires aux utilisateurs, bien que les éléments conflictuels aient été déjà identifiés. De plus, les questionnaires de satisfaction subjective seront proposés après un minimum de corrections des défauts détectés dans le système.

Notre étude a trouvé que les problèmes plus critiques sont dus surtout à l'interface du logiciel Diamant Web.

Notre recherche révèle que les principales difficultés des utilisateurs face aux logiciels cités proviennent d'un traitement du format des données provenant du SIIG incorrect de la part des logiciels. Cette circonstance est défavorisée par l'utilisation intermittente des logiciels, très peu de temps dans chaque session (trois fois par année) ce qui ne permet pas aux utilisateurs de se souvenir de plusieurs détails afin de surmonter les blocages. De plus, la documentation orientée aux fonctions des logiciels ne permet pas aux utilisateurs de reconstruire les pas des tâches ou les pas qui ont été partiellement oubliés. Des éléments conflictuels secondaires se trouvent dans la gestion de grilles depuis Diamant Web.

La solution des problèmes posés consiste à :

1. Ajouter un module dont la fonctionnalité est de vérifier et convertir les données provenant du SIIG automatiquement.

2. Reconcevoir l'interface d'interaction de la gestion de grilles.
3. Concevoir une documentation d'aide et une aide en ligne tous les deux orientées aux tâches de l'utilisateur.

Nos recommandations à court terme consistent à :

1. Inclure une fonction qui permet de faire dans une seule étape et sans intervention de l'utilisateur, la vérification des données provenant du SIIG et la conversion adéquate depuis Diamant Web à Diamant. Cette fonction, en cas d'erreur détectée du SIIG doit fournir au moins des messages explicatifs aux utilisateurs afin qu'ils puissent réviser et corriger les données avant de continuer leur travail.
2. Fournir de la documentation d'aide orientée à la tâche des utilisateurs qui décrivent pas à pas les tâches les plus fréquentes du début à la fin, avec une terminologie appropriée à la tâche afin de rappeler rapidement les détails partiellement oubliés.

Nos recommandations à long terme consistent à :

1. Redéfinir la conception de l'interface d'interaction entre les utilisateurs et le logiciel Diamant Web, surtout dans la création de grilles, la vérification des données provenant du SIIG et leur préparation vis-à-vis leur capture par le logiciel Diamant.
2. Utiliser les techniques orientées à l'utilisabilité, qui brièvement consistent à évaluer l'interaction dès le début de la conception de l'interface proposée avec les utilisateurs réels.
3. Design d'un aide en ligne orientée aux tâches de l'utilisateur. Étant donné le profil de fréquentation de l'utilisateur, une aide en ligne permet un accès rapide aux éléments partiellement oubliés.

ANNEXE C
OUTILS

C.1 Introduction

Cette annexe présente des outils ou des modèles qui montrent :

- comment planifier l'observation de l'utilisateur
- comment construire un aide orientée à la tâche de l'utilisateur, selon un enchaînement des fonctionnalités du logiciel
- comment construire une maquette simple d'interaction pour une tâche particulière

Ces modèles ont été construits pour le système évalué. Toutefois, la dernière partie qui comprend les enquêtes ou questionnaires est générale et applicable à n'importe quelle application logiciel.

C.2 Planification de la session d'observation

La planification d'une session d'observation de l'utilisateur peut se poser de différentes manières selon l'évaluation soit sommative ou formative et selon l'utilisateur soit ancien ou nouveau avec le produit évalué.

Dans ce cas où l'évaluation est sommative, les tâches plus conflictuelles ont été identifiées d'abord et après un rendez-vous avec l'utilisateur a été déterminé afin d'observer ces tâches spécialement. L'utilisateur a été invité à penser en haute voix et l'observateur n'a pas eu aucune intervention pendant la réalisation des tâches. À la fin, l'utilisateur a été invité à rapporter les difficultés qu'il a eues et aussi d'autres problèmes qu'il avait éprouvés auparavant. Un modèle de questions à poser directe ou indirectement est affiché dans le tableau C.1

Dans le cas d'une évaluation formative et/ou pour un utilisateur nouveau, il faut choisir certain tâches et lui inviter à essayer de les réaliser, après une certaine période d'apprentissage s'il est nécessaire. Un modèle de formulaire qui contient les éléments de base à noter est affiché dans le tableau C.2.

C.3 Modèle d'aide orientée-tâche

Après avoir identifié les tâches prioritaires du logiciel Diamant Web, une guide pas à pas d'aide a été construite. Le tableau C.3 affiche les tâches que l'utilisateur réalise avec ce logiciel.

Tableau C.1 – Planification de rapport d'incidents

1	Quelle est (sont) l'erreur (s) le(s) plus fréquente(s) que commettez-vous pendant votre travail ?
2	Quels éléments oubliez-vous d'une session de travail à la suivante ?
3	Quels sont les messages (avertissement et/ou erreur) les plus fréquents que vous recevez de l'interface ?
4	Quel est votre avis par rapport aux couleurs utilisées dans l'interface ?, cela vous aide à faire plus rapidement vos tâches ? Rappelez-vous la signification ?
5	Quels sont les tâches que vous permet d'automatiser le logiciel ?
6	Quelle est la proportion de travail à la main, sur l'ordinateur, qu'estimez-vous devoir faire pour compléter votre tâche ?
7	Avez-vous de suggestions à faire afin d'améliorer l'interface ?
8	De quels éléments désiriez-vous disposer afin d'aller plus vite dans votre travail ?
9	Y-a-t-il des fonctions du logiciel que vous n'utilisez jamais ?

Tableau C.2 – Planification d'observation

Tâche	**Sous-tâche**	**Début**	**Fin**	**Erreurs**	**Messages**	**Commentaires utilisateur**	**Commentaires éval.**
...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...

Dans le cadre d'une étude d'utilisabilité, il est proposé ce modèle de guide pas à pas afin d'instruire aux utilisateurs à la réalisation de tâches complètes. La tâche 1 et ses sous-tâches prioritaires sont présentées ainsi qu'un mini tutoriel pour la tâche 4 de création d'une grille particulière.

La guide complète se trouve dans le fichier "Modèle de Guide Pas à Pas.doc" adjoint à ce travail.

C.3.1 Guide de la tâche 1

Tâche 1 : "projet_horaire.xml" de cours + grille = Horaire cours

Description : Amener le fichier "projet_horaire.xml" de cours dans Diamant Web et préparer la grille et les données pour travailler avec Diamant pour une nouvelle session universitaire

Sous-tâche a : Amener le fichier "projet_horaire.xml" de cours dans Diamant Web pour une nouvelle session et préparer les données pour Diamant

Tableau C.3 – Tâches de l'utilisateur de Diamant Web

N°	Tâche	Description
1	Horaire de cours	Création de l'horaire de cours à partir du fichier projet_horaire.xml et de la grille correspondante
2	Horaire d'examen	Création de l'horaire d'examen à partir du fichier projet_examen.xml et de la grille correspondante
3	Données d'entrée	Vérification du format des données d'entrée
4	Grille d'examen	Création d'une grille d'examen
5	Disponibilité professeur	Modifier la disponibilité d'un professeur
6	Disponibilité faculté/département	Modifier la disponibilité d'une faculté ou d'un département

Sous-tâche b : Préparer la grille pour travailler avec Diamant

Note 1 : Le fichier "projet_horaire.xml" peut avoir n'importe quel nom, mais son extension (3 caractères après le point) doit être xml.

Note 2 : Le fichier "projet_horaire.xml" doit être dans un répertoire de votre ordinateur.

Note 3 : Il est absolument nécessaire (pour l'instant) de valider le format des données de ce fichier avant de commencer. Au contraire, le processus paraîtra marcher dans DiamantWeb mais il sera bloqué dans Diamant.

Suggestions : Créer un nouveau répertoire de travail pour la nouvelle session. Mettre le fichier "projet_horaire.xml" de cours dans ce répertoire. Choisir des noms significatifs à votre travail.

Sous-tâche 1a : Prendre le fichier "projet_horaire.xml" de cours avec Diamant-Web dans une nouvelle session et préparer les données pour Diamant.

Sous-tâche 1b : Préparer la grille pour travailler avec Diamant

Signification des tâches

Sous-tâche 1a : Amener le fichier "projet_horaire.xml" de cours dans Diamant Web pour une nouvelle session et préparer les données pour Diamant. Voir figure C.1

Sous-tâche 1b : Préparer la grille pour travailler avec Diamant. Voir figure C.2

Signification de la tâche complète : Voir figure C.3

01	Ouvrir Diamant Web
02	Cliquer sur disponibilité pour aller à la liste de sessions
03	Cliquer sur ajouter une session
04	Mettre le nom, commentaire et les caractéristiques significatives de votre travail
05	Cliquer sur soumettre pour retourner à la liste de sessions
06	Cliquer sur importer dans la session créée
07	Entrer le nom du fichier project_horaire.xml ou sélectionner du répertoire où il est
08	Cliquer sur associer grille à la session créée pour aller à la liste de grilles à associer
09	Cliquer sur le nom de grille désiré
10	Cliquer sur exporter - sessions. Cela peut demander quelques minutes. Veuillez patienter, s.v.p.
11	Cliquer bouton droit sur télécharger - sessions
12	Cliquer sur enregistrer sous
13	Se placer sur le répertoire de votre travail
14	Changer l'extension du fichier de .txt à .dia
15	Cliquer sur enregistrer

01	Cliquer sur gestion de grilles
02	Cliquer sur exporter de la grille de votre travail (il doit apparaître télécharger, sinon, répéter le clique sur exporter)
03	Cliquer bouton droit sur télécharger (si vous voyez le contenu du fichier dans un format bizarre, vous avez cliqué le bouton gauche, pas problème, fermez la fenêtre et répétez avec le bouton droit)
04	Cliquer sur enregistrer sous
05	Se placer sur le répertoire de votre travail
06	Cliquer sur enregistrer

C.3.2 Mini tutoriel de la tâche 4

Tâche 4 : Création d'une grille d'examen

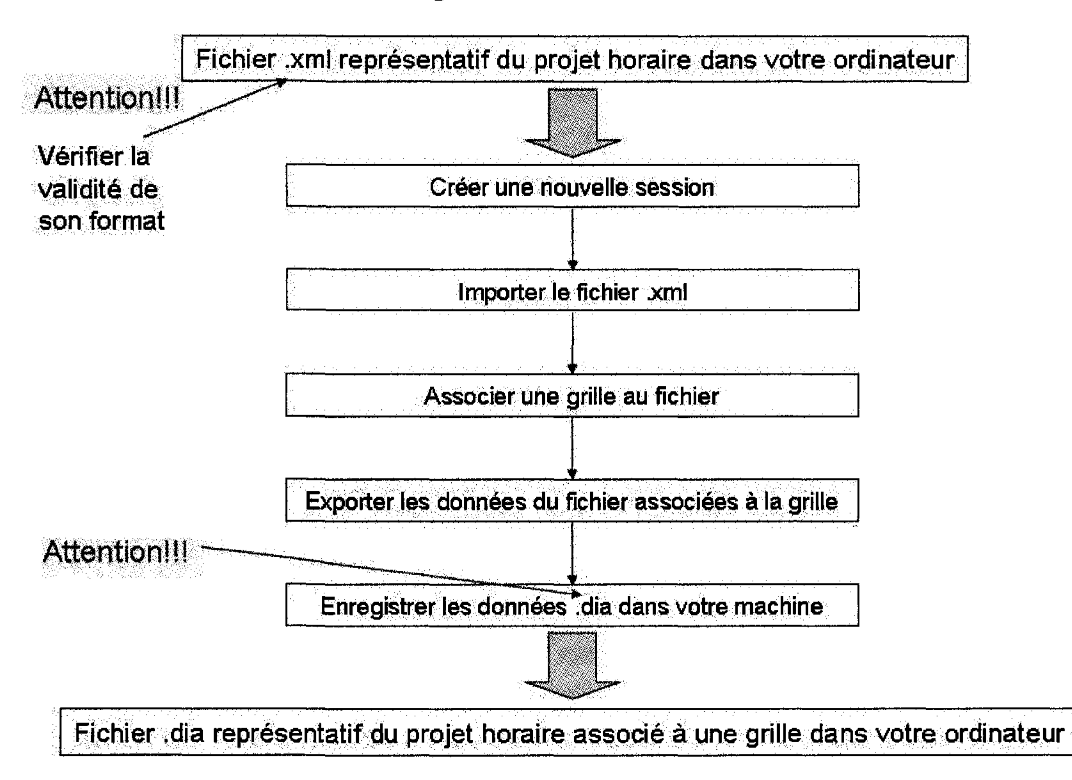
Description : Mini tutoriel qui explique pas à pas comment construire une nouvelle grille d'examen de caractéristiques suivantes :

Grille d'examen de 2 jours (lundi et mardi), 3 séquences (matin, après-midi, soir) chaque jour 2 périodes chaque séquence (9h00-10h00, 10h00-11h00, 14h00-15h00, 15h00-16h00, 18h00-19h00, 19h00-20h00)

Note : Chaque fois qu'il apparaît une fenêtre qui dit "L'opération ..... a été réussie Retourner au ..." Il faut cliquer sur "Retourner"

Voir tableaux C.4 et C.5

Figure C.1 – Tâche01a



C.4 Maquette d'interaction pour la construction de grilles

Étant donné que la construction d'une grille et les changements sur l'horaire afin de réduire ou d'éliminer les conflits ont été identifiés comme des tâches conflictuelles pour l'utilisateur, une maquette d'interface d'interaction nouvelle a été proposée.

Les idées principales sous-jacentes sont :

- Profiter des éléments répétitifs pour les définir dans une seule étape
- Utiliser la représentation graphique de composants éditables et modifiables pour les éléments qui ne sont pas répétitifs ou qui sont des cas particuliers
- Utiliser la représentation graphique et la manipulation directe des objets sur l'écran pour fournir une façon de particularisation plus rapide et intuitive

Les points principaux dans la création d'une nouvelle grille sont affichés dans la figure C.4. Les éléments qui servent à créer une grille et qui sont souvent répétitifs sont affichés dans la figure C.5.

La présentation générale de la grille, les éléments éditables et modifiables pour personnaliser, et les couleurs suggérées apparaissent dans les figures C.6 et C.7.

Figure C.2 – Tâche01b

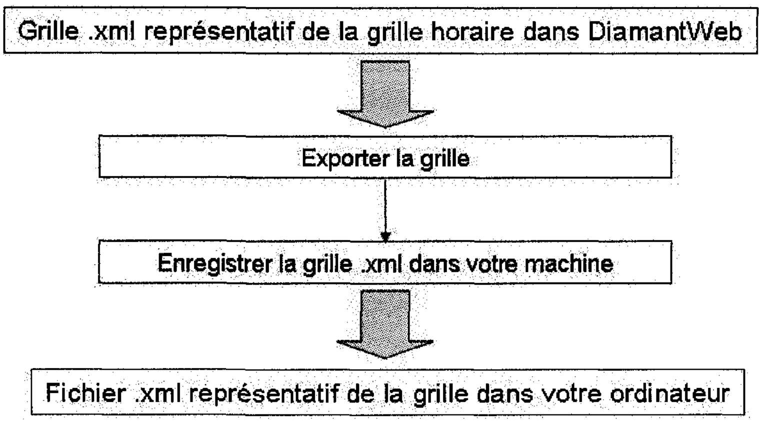
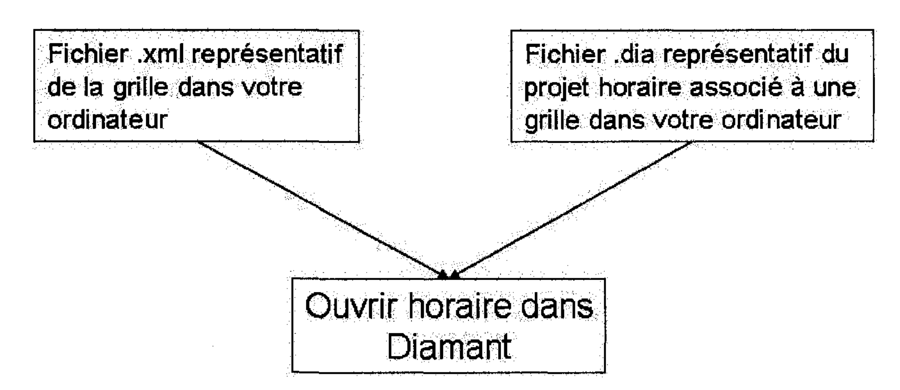


Figure C.3 – Tâche01



En ce qui concerne la présentation de conflits et l'interaction proposée pour permettre leur réduction ou élimination, les points principaux sont montrés dans la C.8. La visualisation rapide des conflits d'un item est exposée dans la figure C.9 où l'utilisation de la manipulation directe permet de déplacer l'élément de la zone de conflit (couleur saumon) à une zone sans conflit (couleur verte).

La présentation générale de conflits se présente dans un seul écran avec la possibilité de voir les détails avec une fonctionnalité "zoom". Cela est affiché dans les figures C.10 et C.12.

Finalement, la figure C.11 montre les couleurs exactes par composants et la codification proposés.

Figure C.4 – Maquette d'interaction : elements principaux de la construction d'une grille

Construction d'une grille

- Présentation de données pour une définition de grille automatique dans une seule étape
- Grille éditable
- Caractéristiques de lignes et colonnes éditables
- Utilisation de la couleur

C.5 Enquête de satisfaction de l'utilisateur

Un outil essentiel d'une évaluation de satisfaction subjective de l'utilisateur est l'enquête ou questionnaire. Cette variable mesurable de la qualité en utilisabilité est la plus importante étant donné que l'utilisateur peut ne pas être satisfait de travailler avec un certain produit logiciel, même si les autres variables (vitesse d'exécution de tâches, productivité, rétention dans la mémoire et période d'apprentissage) se trouvent objectivement dans des limites acceptables.

Un prototype d'enquête apparaît dans le fichier "Prototype_QUS_V01.xls". Ce questionnaire est applicable à n'importe quel produit logiciel utilisé dans un environnement de travail. Ce prototype a été inspiré par [26].

C.6 Enquête de degré de conception centrée sur l'utilisateur

Un outil convenable pour permet aux développeurs de mesurer leur maturité dans la conception centrée sur l'utilisateur et les aspects orientés sur l'utilisabilité, est l'enquête ou questionnaire.

Cet outil donne la possibilité aux développeurs de prendre conscience en ce qui concerne l'utilisabilité et connaître mieux certains aspects qui ont un impact très important sur la qualité de leurs produits.

Un prototype d'enquête est dans le fichier "Test_ConceptionCentréeUsager_V01.xls". Ce questionnaire est applicable à n'importe quel produit logiciel développé. Ce prototype a été inspiré par [27].

Figure C.5 – Construction d'une grille : dessin automatique

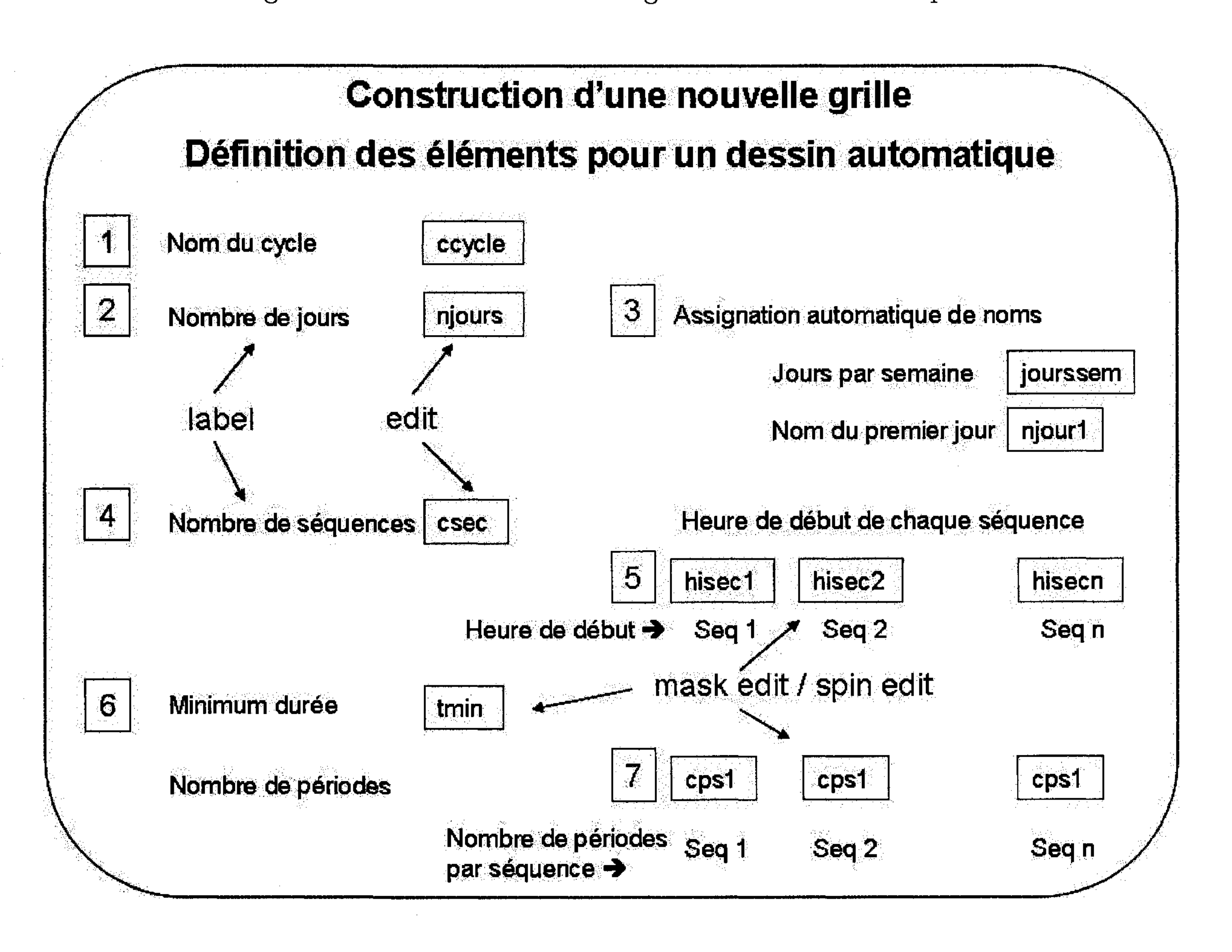


Tableau C.4 – Mini tutoriel de la tâche 4

01	Cliquer sur gestion de grilles du menu principal
02	Cliquer sur ajouter Compléter le nom de la grille (Proof1) et le commentaire
03	Retourner à gestion de grilles
04	Dans gestion de grilles Cliquer sur modifier de la grille créée pour passer au niveau cycle
05	Au niveau cycle Cliquer sur ajouter Compléter le nom du cycle (Proof1Cycle), durée et type
06	Retourner au niveau cycle
07	Au niveau cycle Cliquer sur le nom du cycle (Proof1Cycle) dans le bloc (rectangle) colorié pour passer au niveau jour
08	Au niveau jour Cliquer sur ajouter Compléter le nom du jour (Lundi)
09	Retourner au niveau jour
10	Au niveau jour Cliquer sur ajouter Compléter le nom du jour (Mardi)
11	Retourner au niveau jour
12	Au niveau jour Cliquer sur le nom (Lundi) d'un jour dans le bloc (rectangle) colorié pour passer au niveau séquence
13	Au niveau séquence Cliquer sur ajouter Compléter le nom de la séquence (Lu1Seq1)
14	Cliquer sur ajouter Compléter le nom de la séquence (Lu1Seq2)
15	Cliquer sur ajouter Compléter le nom de la séquence (Lu1Seq3)
16	Cliquer sur le nom (Lu1Seq1) d'une séquence dans le bloc (rectangle) colorié pour passer au niveau période
17	Au niveau période Cliquer sur ajouter Compléter le nom de la période (Lu1Seq1Pe1), l'heure, priorité
	Cliquer sur ajouter Compléter le nom de la période (Lu1Seq1Pe2), l'heure, priorité
18	Retourner au niveau séquence

Tableau C.5 – Mini tutoriel de la tâche 4 (continuation)

19	Cliquer sur le nom de la séquence suivante (Lu1Seq2)
20	Cliquer sur ajouter Compléter le nom de la période (Lu1Seq2Pe3), l'heure, priorité
21	Cliquer sur ajouter Compléter le nom de la période (Lu1Seq2Pe4), l'heure, priorité
22	Retourner au niveau séquence
23	Cliquer sur le nom de la séquence suivante (Lu1Seq3)
24	Cliquer sur ajouter Compléter le nom de la période (Lu1Seq3Pe5), l'heure, priorité
25	Cliquer sur ajouter Compléter le nom de la période (Lu1Seq3Pe6), l'heure, priorité
26	Retourner au niveau jour
27	Cliquer sur le numéro du jour (1)
28	Cliquer sur copier
29	Cliquer sur le numéro du jour (2). Il va a apparaît Lu-¿Ma
	Pour changer les noms de jour, séquences et périodes du jour qui vient d'être copié :
30	Cliquer sur le numéro correspondant dans le bloc (rectangle) colorié
31	Cliquer sur éditer
32	Changer le nom qui apparaît
	En tout temps Pour passer d'un niveau supérieur a un niveau inférieur Cliquer sur le nom correspondant dans le bloc (rectangle) colorié
	En tout temps Pour afficher Cliquer sur afficher, pour aficher la grille qui est en train d'être construite
	En tout temps Pour effacer Cliquer sur le numéro correspondant dans le bloc (rectangle) colorié. Vérifier visuellement que ce rectangle change sa couleur Cliquer sur effacer

Figure C.6 – Présentation d'une grille : elements éditables et manipulation directe

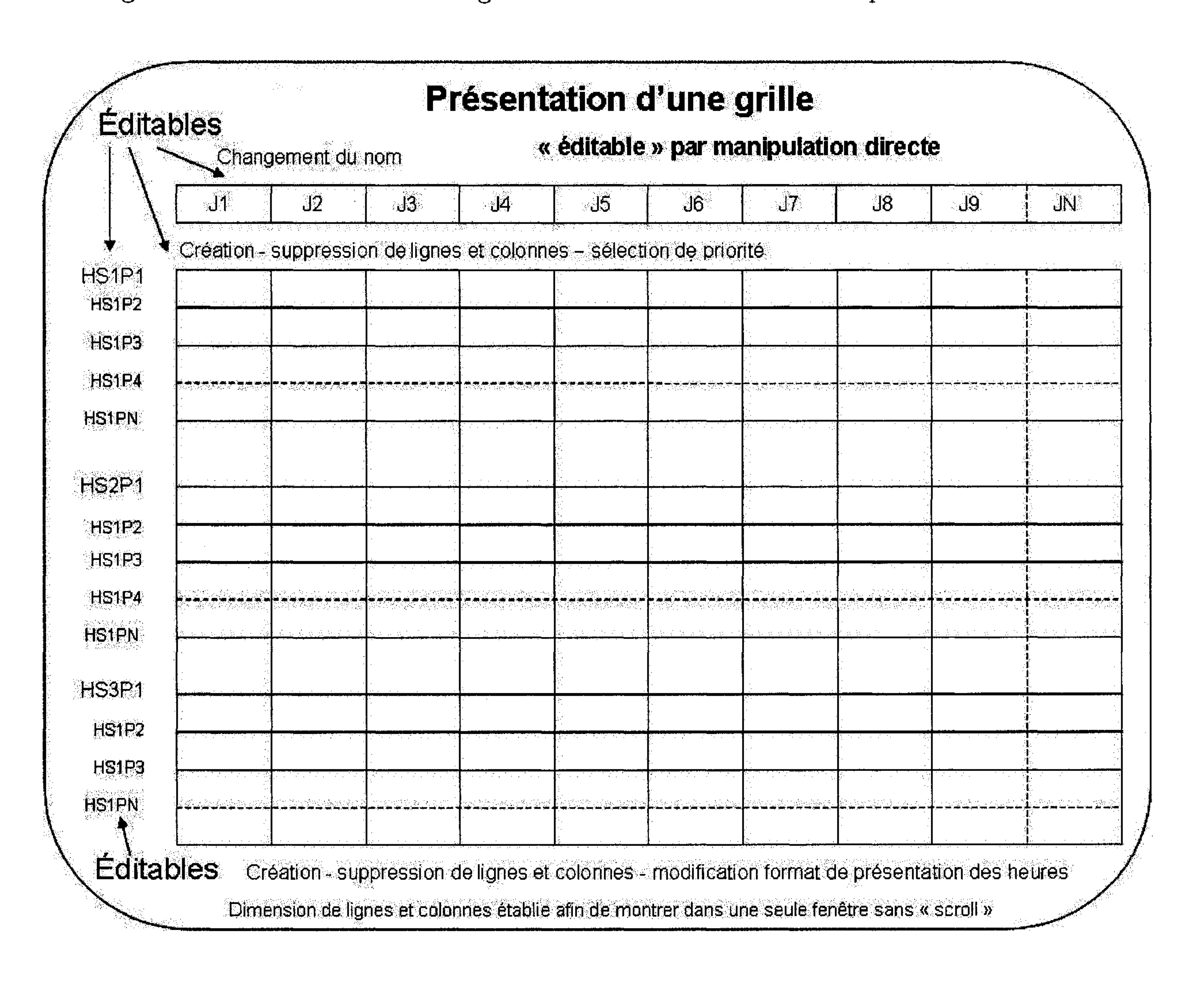


Figure C.7 – Présentation d'une grille : Utilisation de la couleur

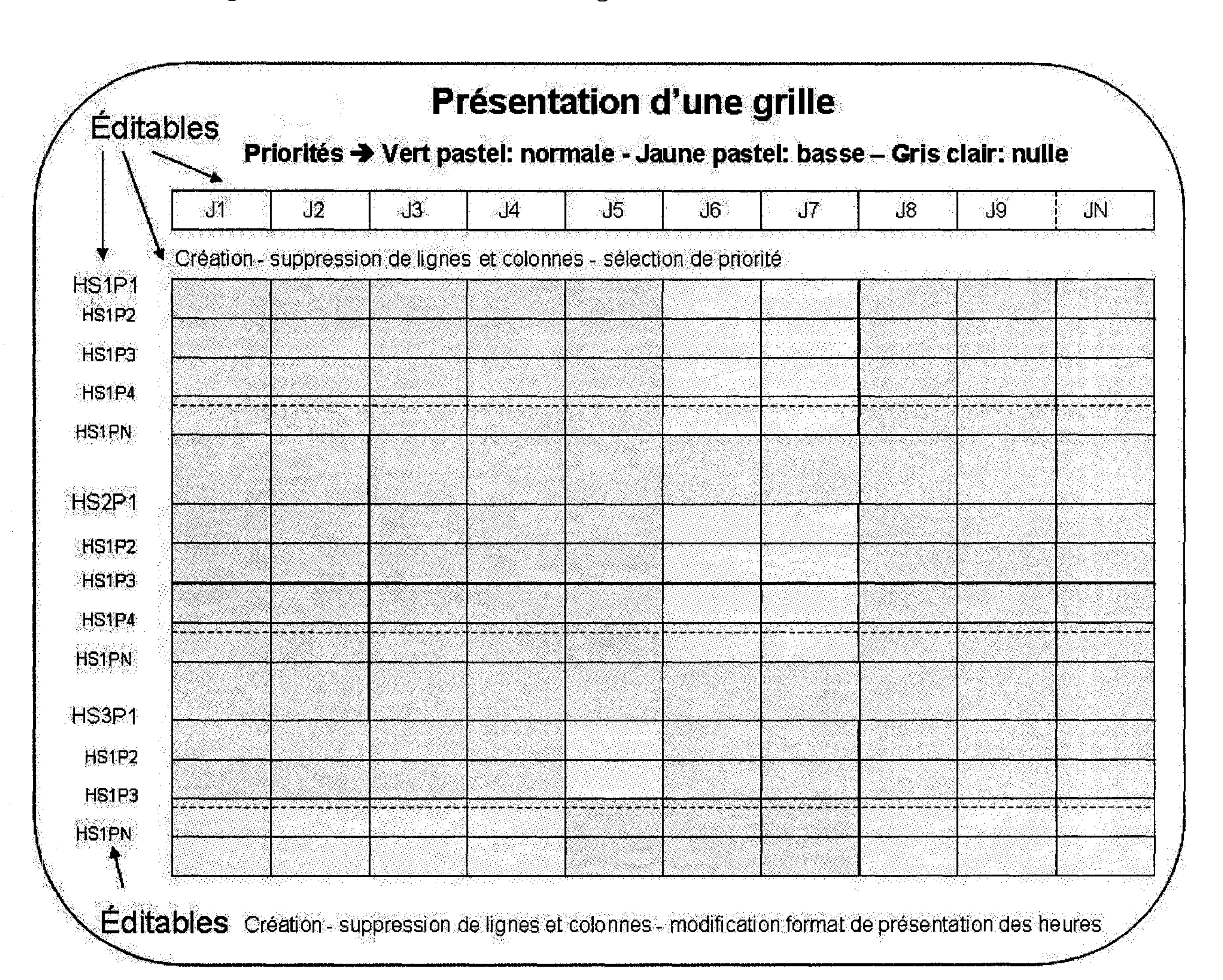


Figure C.8 – Présentation d'un horaire : Conflits

Présentation de l'horaire

- Conflits globaux
- Conflit local
- Caractéristiques de lignes et colonnes éditables
- Utilisation de la couleur

Figure C.9 – Présentation des conflits par item : manipulation directe

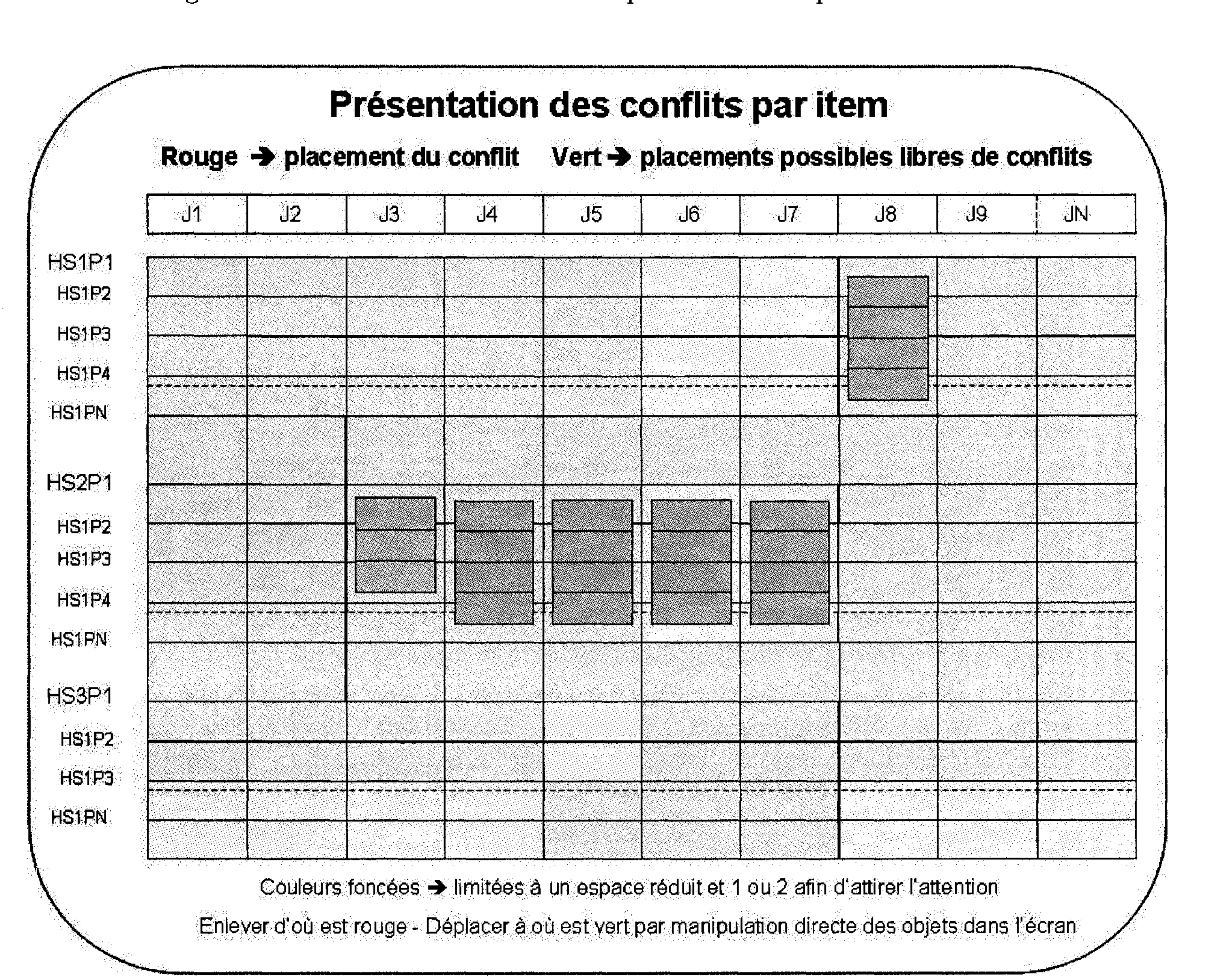


Figure C.10 – Présentation des conflits : zoom

Présentation général de conflits

Proportion de conflit en rouge - Magnifier pour visualiser détails

J1 J2 J3 J4 J5 J6 J7 J8 J9 JN

Création - suppression de lignes et colonnes - sélection de priorité

HS1P1
HS1P2
HS1P3
HS1P4
HS1PN

HS2P1
HS1P2
HS1P3
HS1P4
HS1PN

HS3P1
HS1P2
HS1P3
HS1PN

2 Étudiants
5 Enseignants
3 Locaux

1 Étudiants
0 Enseignants
0 Locaux

3 Étudiants
0 Enseignants
1 Locaux

0 Étudiants
0 Enseignants
0 Locaux

Dimension de lignes et colonnes escalées afin de montrer dans une seule fenêtre
sans « scroll » et avec « zoom »

Figure C.11 – Codification par les couleurs

Codification par les couleurs

Pour la grille

Priorité normal	Vert pastel	231 255 231
Priorité basse	Jaune pastel	255 255 217
Priorité nulle	Gris clair	243 243 243

Pour l'horaire

Sans conflit	Vert	255 203 167
Avec conflit	Saumon	168 226 197

Codification en accord avec la signification naturelle

Permission ➔Vert Interdiction ➔ Rouge Attention ➔Jaune

Figure C.12 – Détails de conflits

Détail de conflits

2 Étudiants 5 Enseignants 3 Locaux	1 Étudiants 0 Enseignants 0 Locaux
3 Étudiants 0 Enseignants 1 Locaux	0 Étudiants 0 Enseignants 0 Locaux

Bibliographie

[1] C. Barnum. *Usability Testing and Research.* Allyn and Bacon, 2001.

[2] K. Beck and C. Andres. *Extreme Programming Explained : Embrace Change.* Addison Wesley, 2004.

[3] K. Beck and M. Fowler. *Planning Extreme Programming.* Addison Wesley, 2000.

[4] R. Bias and D. Mayhew. *Cost Justifying Usability.* Morgan Kaufmann, 2005.

[5] S. Charlton and T. O'Brien. *Handbook of Human Factors Testing and Evaluation.* John Wiley and Sons, 2001.

[6] A. Cockburn. *Agile Software Development.* Addison Wesley, 2002.

[7] A. Dix, J. Finlay, G. Abouwd, and R.Beale. *Human-Computer Interaction.* Prentice Hall, 1998.

[8] J. Dumas and J. Redish. *A Practical Guide of Usability Testing.* Intellect, 1999.

[9] Michael Hahnel, Nico Hamacher, and Karl-Friedrich Kraiss. Usability evaluation of interactive systems using trevis. *WWDU 2002, 6th International Scientific Conference on Work with Display Units - World Wide Work*, 2002.

[10] W. Hansen. User engineering principles for interactive systems. *American Federation of Information Processing Societies Conference Proceedings*, 39 :523–532, 1971.

[11] Morten Hertzum and Niels Ebbe Jacobsen. The evaluator effect : A chilling fact about usability evaluation methods. *International Journal of Human-Computer Interaction*, 13(4) :421–443, 2001.

[12] Morten Hertzum, Niels Ebbe Jacobsen, and Rolf Molich. Usability inspections by groups of specialists : Perceived agreement in spite of disparate observations. *ACM CHI 2002 Conference ACM Press*, pages 662–663, April 2002.

[13] Deborah Hix, J. Edward Swan, Joseph L. Gabbard, Mark A. Livingston, Tobias H. Höllerer, Simon J. Julier, Yohan Baillot, and Dennis Brown. A cost-effective usability evaluation progression for novel interactive systems. *Proceedings of the Hawaii International Conference on System Sciences HICSS-37*, January 2004.

[14] A. Hunt and D. Thomas. *The Pragmatic Programmer : From Journeyman to Master.* Addison Wesley, 1999.

[15] Melody Y. Ivory and Marti A. Hearst. The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys*, 33(4) :470–516, December 2001.

[16] Laurie Kantner, Deborah Hinderer Sova, and Stephanie Rosenbaum. Alternative methods for field usability research. *SIGDOC 2003 Proceedings, published by Association for Computing Machinery, Inc.*, 2003.

[17] M. Kuniavsky. *Observing the User Experience : A Practitioner's Guide to User Research.* Morgan Kaufmann, 2003.

[18] C. Larman. *Agile and Iterative Development : A Manager's Guide.* Addison Wesley, 2004.

[19] J. Nielsen. *Usability Engineering.* Morgan Kaufmann, 1994.

[20] J. Nielsen. Search : Visible and simple. *Jakob Nielsen Alertbox*, May 2001.

[21] J. Nielsen and H. Loranger. *Prioritizing Web Usability.* New Riders Publishing, 2000.

[22] Philippe Palanque, Regina Bernhaupt, Ronald Boring, and Chris Johnson. Testing interactive software : A challenge for usability and reliability. *CHI 2006*, April 2006.

[23] J. Preece, Y. Rogers, and H. Sharp. *Interaction Design : Beyond Human-Computer Interaction.* John Wiley and Sons, Inc., 2002.

[24] J. Rubin. *Handbook of Usability Testing : How to Plain, Design and Conduct Effective Tests.* John Wiley and Sons, 1994.

[25] J. Scholtz. Usability evaluation. *National Institute of Standards and Technology*, 2003.

[26] B. Shneidermann and C. Plaisant. *Designing the User Interface : Strategies for Effective Human-Computer Interaction.* Pearson Education, Inc., 2005.

[27] D. Travis. *E-Commerce Usability.* Taylor and Francis, 2002.